Implementation and Evaluation of Nonparametric Regression Procedures for Sensitivity Analysis of Computationally Demanding Models

Curtis B. Storlie[a], Laura P. Swiler[b], Jon C. Helton[b] and Cedric J. Sallaberry[b],

[a]Department of Mathematics & Statistics, University of New Mexico,
storlie@stat.unm.edu
[b]Sandia National Laboratories, Albuquerque, NM 87185-0776 USA

November 18, 2008

Abstract

The understanding of many physical and engineering problems involves running complex computational models (computer codes). With problems of this type, it is important to understand the relationships between the input variables (whose values are often imprecisely known) and the output. The goal of sensitivity analysis (SA) is to study this relationship and identify the most significant factors or variables affecting the results of the model. In this presentation, an improvement on existing methods for SA of complex computer models is suggested for use when the model is too computationally expensive for a standard Monte-Carlo analysis. In these situations, a meta-model or surrogate model can be used to estimate the necessary sensitivity index for each input. A sensitivity index is a measure of the variance in the response that is due to an input. Most existing approaches to this problem either do not work well with a large number of input variables and/or they ignore the error involved in estimating a sensitivity index. Here, a new approach to sensitivity index estimation using meta models and bootstrap confidence intervals is proposed that appears to provide satisfactory solutions to these drawbacks. Further, an efficient yet very effective approach to incorporate this methodology into an actual SA is presented. Several simulation and real data examples illustrate the utility of this approach. This framework can be easily extended to uncertainty analysis as well.

1 Introduction

The analysis of many physical and engineering phenomena involves running complex computational models (computer codes). It is almost universally accepted that the sensitivity analysis (SA) and uncertainty analysis (UA) of these complex models is an important and necessary component to overall analyses [1, 2, 3, 4, 5]. The purpose of SA is to identify the most significant factors or variables affecting the model predictions. The purpose of UA is to quantify the uncertainty in analysis results due to the uncertainty in the inputs. A computational model that sufficiently represents reality is often very costly in terms of run time. Thus, it is important to be able to characterize model uncertainty and perform SA with a limited number of model runs.

In this report, we suggest an effective procedure for SA of such expensive computer models using meta-models and variance based sensitivity measures. This approach has several advantages over existing procedures: (i) efficient use of computational resources, (ii) effective handling of a very large number of input variables, and (iii) generation of confidence interval estimates of sensitivity and/or uncertainty measures.

In general, we will consider complex computer models of the form

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon}, \tag{1.1}$$

where $\boldsymbol{y} = (y_1, \ldots, y_q)$ is a vector of outputs, $\boldsymbol{x} = [x_1, x_2, \ldots, x_p]$ is a vector of imprecisely known inputs, and $\boldsymbol{\varepsilon}$ is a vector of errors (usually small) incurred by the numerical method used to solve for $\boldsymbol{y}$. For example, $\boldsymbol{\varepsilon}$ could result from chaotic behavior introduced by a stopping criterion where input configurations arbitrarily close to one another can fail to achieve convergence in the same number of iterations. Although analyses for real systems almost always involve multiple output variables as indicated above, the following discussions assume that a single real-valued result of the form $y = f(\boldsymbol{x}) + \varepsilon$ is under consideration. This simplifies the notation and the results under discussion are valid for individual elements of $\boldsymbol{y}$.

The model f can be quite large and involved (e.g., a system of nonlinear partial differential equations requiring numerical solution or possibly a sequence of complex, linked models as is the case in a probabilistic risk assessment for a nuclear power plant [6] or a performance assessment for a radioactive waste disposal facility [7]); the vector $\boldsymbol{x}$ of analysis inputs can be of high dimension and complex structure (i.e., several hundred variables, with individual variables corresponding to physical properties of the system under study or perhaps to designators for alternative models).

The uncertainty in each element of $\boldsymbol{x}$ is typically characterized by a probability distribution. Such distributions are intended to numerically capture the existing knowledge about the elements of $\boldsymbol{x}$ and are often developed through an expert review process. See [8] and [9] for more on the characterization of input variable uncertainty. After the characterization of this uncertainty, a number of approaches to SA are available, including differential analysis, variance decomposition procedures, Monte Carlo (sampling-based) analysis, and evidence theory representations [10]. Variance decomposition is perhaps the most informative and intuitive means with which to summarize the uncertainty in analysis output resulting from uncertainty in individual input variables. This procedure uses measures such as

$$s_j = \frac{\mathrm{Var}(\mathrm{E}[f(\boldsymbol{x}) \mid x_j])}{\mathrm{Var}(f(\boldsymbol{x}))} \tag{1.2}$$

and

$$T_j = \frac{\mathrm{E}(\mathrm{Var}[f(\boldsymbol{x}) \mid \boldsymbol{x}_{(-j)}])}{\mathrm{Var}(f(\boldsymbol{x}))} = \frac{\mathrm{Var}(f(\boldsymbol{x})) - \mathrm{Var}(\mathrm{E}[f(\boldsymbol{x}) \mid \boldsymbol{x}_{(-j)}])}{\mathrm{Var}(f(\boldsymbol{x}))}, \tag{1.3}$$

where $\boldsymbol{x}_{(-j)} = \{x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_p\}$, to quantify this uncertainty. The use of these measures is reviewed in [8]. The quantity s_j corresponds to the proportion of the uncertainty in y that can be attributed to x_j alone, while T_j corresponds to

the total uncertainty that can be attributed to x_j and its interactions with other variables. These calculations require the evaluation of p-dimensional integrals which are typically approximated via Monte Carlo sampling on the joint distribution of $\boldsymbol{x}$. Unfortunately, this is too computationally intensive to be feasible for most complex computer models.

An alternative procedure to the direct evaluation of T_j and similar measures is to use a meta-model (or surrogate) for f to perform the necessary model evaluations ([11], [12]). A meta-model, denoted $\hat{f}$, is much simpler in form and faster to evaluate than the actual computer model. This approach involves taking a sample of size n from the joint distribution (e.g. a simple random sample or Latin hypercube sample [13] and [14]) and evaluating the actual computer model, f, at each of the n design points. The data can then be used to create a meta-model for f. It is assumed that n is a fairly modest number of model evaluations, but large enough to allow for a flexible meta-model estimation. The most commonly used method for function estimation is linear regression which has been used with much success for SA when the underlying function is approximately linear. However, it is often the case that linear regression can fail to appropriately identify the effects of the elements of $\boldsymbol{x}$ on y when nonlinear relations are present. Rank regression works very well to identify the strength of relationships between inputs and output in nonlinear situations as long the relationships between inputs and output are approximately monotonic [9] and [15]. However, rank regression does not provide a meta-model as the resultant regression model does not directly provide useful output predictions at new $\boldsymbol{x}$ locations. In nonlinear situations, nonparametric regression methods can be used to achieve a better approximation than can be obtained with linear regression procedures [12].

In this presentation, we describe several modern nonparametric regression methods and compare their performance in calculating sensitivity measures. We also present a general approach to calculating confidence intervals for these measures. This allows a practitioner to account for the variability (i.e. sampling based error) involved in the assessment of variable importance. This presentation continues the investigation of the use of nonparametric regression procedures in SA initiated in [12] by presenting (i) comparisons of several state of the art meta-models, (ii) more relevant and precisely defined sensitivity measures, (iii) confidence intervals for these measures, and (iv) a general method for fast yet effective sensitivity analysis of complex models.

In Section 2 we describe how to use a flexible meta-model to calculate sensitivity measures and associated confidence intervals. We then discuss some of the more useful nonparametric regression procedures available to fit meta-models in Section 3. A simulation study to illustrate the properties of the proposed methodology is given in Section 4. Section 5 describes an efficient procedure for implementation and gives an example of this approach in practice. Finally a concluding discussion is given in Section 6.

2 Calculating Sensitivity Measures

Assume for the following that a design has been generated for the uncertain inputs (either fixed or random) and the model has been evaluated at the design points. In this section we consider the calculation of sensitivity indexes for two cases: (i) A linear and/or rank regression is used to fit this data or (ii) A more flexible model is needed.

2.1 Sensitivity Measures for Linear and Rank Regression

If the fit from linear or rank regression is adequate, then we recommend using the familiar approach of calculating Standardized Regression Coefficients (SRCs) and Partial Correlation Coefficients (PCCs) or the analog of these quantities for rank data as described in [9, 16], and [17]. The main reasons for this are (i) These quantities are familiar and simple to understand and (ii) they are very fast and easy to calculate.

If the fit from both the linear and rank regressions is inadequate, then we recommend calculating the quantities described in Section 2.2 below. The definition of an "adequate" fit is of course somewhat arbitrary. We recommend the following rule: if the R^2 value of the model is greater than a cut-off value then the fit is adequate. In practice, the appropriate cut-off value to use is clearly problem dependent. We discuss this issue further in Section 5.

2.2 Sensitivity Measures for More Flexible Models

When the fits from a linear and rank regression are inadequate, a more flexible meta-model can be used. The discussion of meta-model choice is delayed until Section 3. It suffices for the discussion here to assume we are using an appropriate meta-model to approximate the computer model. In this case, there are two basic types of sensitivity measures that we will consider. The first is the T_j given in Eq. (1.3), which was proposed by Homma and Saltelli [18]. This is generally a very good single number summary of the overall importance of an input variable. It has the interpretation of "the total proportion of the uncertainty in y due to x_j (including $x_j's$ interaction with other variables". It is important to stress that we will be using a meta-model, $\hat{f}$, to carry out the necessary calculations to obtain T_j. Hence, the result is really an estimate of the true T_j given by

$$\hat{T}_j = \frac{\mathrm{E}(\mathrm{Var}[\hat{f}(\boldsymbol{x}) \mid \boldsymbol{x}_{(-j)}])}{\mathrm{Var}(\hat{f}(\boldsymbol{x}))}. \tag{2.1}$$

The second measure we will use is the stepwise variance contribution, denoted S_j. Notice that we use uppercase S_j to distinguish from the main effect variance contribution, s_j, defined in Eq. (1.2). The quantity S_j is motivated by stepwise model fitting. When building a model in a stepwise fashion, it is informative to observe the increase in R^2 (uncertainty in y accounted for by the model) for each variable that

enters the model. The meta-model used does not need to be constructed in a stepwise fashion in order to use S_j. We can still obtain such stepwise measures by calculating the following quantities from the fitted model.

Define the first variable to "enter" the model by x_j for the j that maximizes

$$U_{1,j} = \frac{\text{Var}[\text{E}(f(\boldsymbol{x})|x_j)]}{\text{Var}(f(\boldsymbol{x}))}. \tag{2.2}$$

Let $a_1 = \arg\max_j U_{1,j}$ so that x_{a_1} is the input variable that maximized $U_{1,j}$. The second variable to "enter" is then defined by the maximizer of

$$U_{2,j} = \frac{\text{Var}[\text{E}(f(\boldsymbol{x})|x_{a_1}, x_j)]}{\text{Var}(f(\boldsymbol{x}))}. \tag{2.3}$$

Similarly, define x_{a_2} to be the variable that maximizes $U_{2,j}$. In general, the k^{th} variable to "enter" the model corresponds to the maximizer of

$$U_{k,j} = \frac{\text{Var}[\text{E}(f(\boldsymbol{x})|x_{a_1}, x_{a_2}, ..., x_{a_{k-1}}, x_j)]}{\text{Var}(f(\boldsymbol{x}))}. \tag{2.4}$$

Now, define

$$S_{a_k} = U_{k,a_k} \text{ for } k = 1, \ldots, p. \tag{2.5}$$

The differences between successive S_j provide a measure of the incremental increase in the proportion of the uncertainty explained by including the uncertainty of the j^{th} variable in a stepwise manner. This has a lot of intuitive appeal since practitioners are familiar with the concept of stepwise contribution when adding a variable to an existing model. The quantities S_j and T_j parallel the relationship between Type I sums of squares and Type III sums of squares for regression models in the popular SAS software ([19]).

Finally, define $\hat{U}_{k,a_k}$, $k = 1, \ldots, p$, by replacing f with $\hat{f}$ in Eqs. (2.2) - (2.4) and let the estimate of S_j be given as

$$\hat{S}_{a_k} = \hat{U}_{k,a_k} \text{ for } k = 1, \ldots, p. \tag{2.6}$$

The quantities $\hat{S}_j$ and $\hat{T}_j$ in Eqs. (2.6) and (2.1) can be calculated via Monte Carlo sampling over the $\boldsymbol{x}$ distribution; see p. 178 of [8] for details. This requires many thousands of model evaluations to obtain accurate Monte Carlo approximations. This is quite feasible however, because $\hat{f}$ is much faster to evaluate than f.

2.3 Confidence Intervals

The biggest advancement included in this work is the introduction of confidence intervals (CIs) for sensitivity measures estimated from meta-models. The use of meta models for estimating sensitivity measures can be much more accurate than the use of standard Monte Carlo methods for estimating these measures with small to

moderate sample sizes. However, exactly how accurate these estimates are for a given sample size is of course problem dependent. It is very important to know how much confidence we can have in our importance measures and rankings for the individual input variables.

By definition, a $100(1-\alpha)\%$ CI for T_j should contain the true value of T_j $100(1-\alpha)\%$ of the time under repeated experimentation. In our case, the experiment entails taking a sample of size n from the $\boldsymbol{x}$ distribution, evaluating the computer model at these n design points, and then using these n values to create a confidence interval for T_j. If we repeat this experiment say 1,000 times, we would expect 950 of the resultant 1,000 95% CIs for T_j to contain the true value of T_j.

Such a CI for T_j can be calculated using a bootstrap approach which will be described shortly. It might also be of interest to obtain CIs for other quantities as well. These could be calculated using the same approach described below. It is important to recognize that the confidence sets formed from the popular Gaussian process (or Kriging) models are not confidence intervals by the definition above. These are really Bayesian Credible sets as discussed in Section 3.5, which are a different entity. This is not to say that Gaussian process models are not useful; in fact, they can be quite useful as demonstrated in Section 4. We begin our discussion of bootstrap CIs by reviewing the basic bootstrap.

Review of Bootstrap CIs

Let Y denote a random variable (RV) characterized by a cumulative distribution function (CDF) F. Suppose we observe a random sample

$$\boldsymbol{Y} = (Y_1, \ldots, Y_n) \tag{2.7}$$

from F. Now let

$$\hat{\theta} = \hat{\theta}(\boldsymbol{Y}) \tag{2.8}$$

be an estimate for some parameter of interest

$$\theta = \theta(F). \tag{2.9}$$

If we knew the distribution of $(\hat{\theta} - \theta)$, we could find $z_{\alpha/2}$ and $z_{1-\alpha/2}$ such that

$$\Pr(z_{\alpha/2} \leq \hat{\theta} - \theta \leq z_{1-\alpha/2}) = 1 - \alpha, \tag{2.10}$$

where $\Pr(A)$ is the probability of an event A. Then,

$$(\hat{\theta} - z_{1-\alpha/2}, \hat{\theta} - z_{\alpha/2}) \tag{2.11}$$

is a $(1-\alpha)100\%$ CI for θ. Generally we don't know the distribution of $(\hat{\theta} - \theta)$ but we can approximate it with a bootstrap distribution [20], [21].

The main idea behind the bootstrap procedure is to use an estimated CDF, F^*, in place of F. There are many ways to obtain F^* (e.g. use the empirical CDF or

somehow parametrize F so it belongs to a family, such as Gaussian, exponential, etc., and estimate the parameters of that family). Once we obtain F^*, we can mimic the data generating process that produced $\boldsymbol{Y}$. That is, we can draw a sample of size n from F^*, denoted as $\boldsymbol{Y}^* = (Y_1^*, \ldots, Y_n^*)$. Here, $\boldsymbol{Y}^*$ is called a *bootstrap sample* to emphasize that it is a sample from the estimated CDF, F^*.

The most familiar bootstrap procedure is that which samples the data with replacement to obtain the bootstrap sample $\boldsymbol{Y}^*$. This is equivalent to taking a random sample from the empirical CDF (i.e. using $F^*(y) = 1/n \sum_{i=1}^n I_{(-\infty, Y_i]}(y)$ where $I_A(x)$ is the indicator function, $I_A(x) = 1$ if $x \in A$ and 0 otherwise). If the empirical CDF is used for F^* the following procedure is called the *nonparametric bootstrap*. However, any estimate of F can be used to draw a bootstrap sample. If the estimate F^* is obtained by some other means than the empirical CDF, the following procedure is called the *parametric bootstrap*.

Once a bootstrap sample $\boldsymbol{Y}^*$ is created, we could then calculate $\hat{\theta}$ for the the bootstrap sample, that is

$$\hat{\theta}^* = \hat{\theta}(\boldsymbol{Y}^*) \tag{2.12}$$

where the * is to emphasize that $\hat{\theta}^*$ is an estimate of θ which came from a bootstrap sample. Once F^* is obtained the value of the parameter θ for the CDF F^*,

$$\theta^* = \theta(F^*), \tag{2.13}$$

is also known (or at least can be calculated).

Recall, the goal is to approximate the $\alpha/2$ and $1-\alpha/2$ quantiles ($z_{\alpha/2}$ and $z_{1-\alpha/2}$) of the $Z = (\hat{\theta} - \theta)$ distribution. With the bootstrap procedure, these quantities are approximated using the distribution of $Z^* = (\hat{\theta}^* - \theta^*)$. Denote the $\alpha/2$ and $1-\alpha/2$ quantiles of the Z^* distribution as $z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$, respectively. Sometimes it is possible to calculate $z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$ analytically. If so, the bootstrap CI can be given as

$$(\hat{\theta} - \tilde{z}^*_{1-\alpha/2}, \hat{\theta} - \tilde{z}^*_{\alpha/2}). \tag{2.14}$$

which is an approximation to Eq. (2.11).

More often, it is necessary to approximate $z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$ using bootstrap sampling. That is, generate M samples $\boldsymbol{Y}^*_k$, $k = 1, \ldots, M$, from the F^* distribution), each time calculating

$$\hat{\theta}^*_k = \hat{\theta}(\boldsymbol{Y}^*_k). \tag{2.15}$$

Now order the $\hat{\theta}^*_k$ from smallest to largest. That is, let $\hat{\theta}^*_{(j)} = \hat{\theta}^*_k$ for some $k = 1, \ldots, M$, and $\hat{\theta}^*_{(1)} \leq \hat{\theta}^*_{(2)} \leq \cdots \leq \hat{\theta}^*_{(M)}$, Then, let

$$\tilde{z}^*_q = \hat{\theta}^*_{([qM])} - \theta^*, \tag{2.16}$$

where $[qM]$ is the closest integer to qM, for $0 \leq q \leq 1$. The bootstrap CI is then given by

$$(\hat{\theta} - \tilde{z}^*_{1-\alpha/2}, \hat{\theta} - \tilde{z}^*_{\alpha/2}). \tag{2.17}$$

This is an approximation to Eq. (2.11). Notice that there are two sources of approximation error involved in creating the CI in Eq. 2.17, (i) the error involved in approximating $z_{\alpha/2}$ and $z_{1-\alpha/2}$ with the corresponding quantities from the bootstrap distribution, $z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$ and (i) the sampling error involved in further approximating $z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$ with $\tilde{z}^*_{\alpha/2}$ and $\tilde{z}^*_{1-\alpha/2}$. In concept the error represented in (ii) can be made arbitrarily small by making the number of the bootstrap samples, M, large. The error described in (i) can only be reduced by increasing the size of the original sample, $\boldsymbol{Y}$. Hence the error in (ii) is generally small or negligible compared to the error described in (i).

Under certain regularity conditions on F, F^* and $\hat{\theta}$, this procedure will produce asymptotically correct CIs (p. 37-39, [20]). The performance of these CIs in practice is determined by how closely the distribution of $Z^* = (\hat{\theta}^* - \theta^*)$ matches that of $Z = (\hat{\theta} - \theta)$. Of course, the goal is to have the distribution of Z^* resemble the distribution of Z as closely as possible. Sometimes another function of $\hat{\theta}$ and θ instead of $Z = \hat{\theta}^* - \theta^*$ can be preferable to accomplish this. In fact, any function of θ and $\hat{\theta}$, $g(\hat{\theta}, \theta)$, that can be solved for θ for a fixed $\hat{\theta}$ can be used to calculate a bootstrap CI. A common alternative to the basic bootstrap is to standardize by dividing by a variance estimate, e.g.

$$U = \frac{(\hat{\theta} - \theta)}{\sqrt{\widehat{\mathrm{Var}}(\hat{\theta})}} \tag{2.18}$$

and the CI is given as

$$\left(\hat{\theta} - \tilde{u}^*_{1-\alpha/2}\sqrt{\widehat{\mathrm{Var}}(\hat{\theta})}, \hat{\theta} - \tilde{u}^*_{\alpha/2}\sqrt{\widehat{\mathrm{Var}}(\hat{\theta})}\right) \tag{2.19}$$

where the $\tilde{u}^*_{\alpha/2}$ and $\tilde{u}^*_{1-\alpha/2}$ are calculated using

$$\tilde{u}^*_q = \frac{\hat{\theta}^*_{([qM])} - \theta^*}{\sqrt{\widehat{\mathrm{Var}}(\hat{\theta}^*)}} \tag{2.20}$$

in a similar manner to Eq. (2.16). This is called the t bootstrap because of the similarity to the t statistic. The t bootstrap is known to speed up the rate of convergence and perform better than the standard bootstrap (using Z instead of U) in practice in many instances [20].

Bootstrap CIs for T_j

There are many ways to construct bootstrap CIs for the T_j in our situation. Here we describe one form of the parametric bootstrap with which we have had reasonable results. First, we construct an appropriate meta-model using the sample $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ to obtain an estimate, $\hat{f}$, of the computer model, f. We use

the meta model, $\hat{f}$, as though it is the actual computer model to carry out the calculation of $\hat{T}_j$ as defined in Eq. (2.1). For the rest of the discussion, assume that j is fixed. However, in practice the indicated calculation would be performed for $j = 1, \ldots, p$.

The goal is to obtain a CI for T_j; hence, we will discuss a procedure to obtain a bootstrap distribution of

$$Z = \hat{T}_j - T_j. \tag{2.21}$$

A CI for T_j is then given by

$$\left(\hat{T}_j - z^*_{1-\alpha/2}\ ,\ \hat{T}_j - z^*_{\alpha/2}\right), \tag{2.22}$$

where $z^*_{1-\alpha/2}$ and $z^*_{\alpha/2}$ are determined from the bootstrap distribution of Z which is obtained as described below.

In this case, the data consists of values for $(\boldsymbol{x}, y)$; thus, F^* is a joint distribution for the variables $(\boldsymbol{x}, y)$. We do not actually need to define F^* explicitly as we only need to be able to draw a bootstrap sample, $(\boldsymbol{x}^*_i, y^*_i)$, $i = 1, \ldots, n$, from F^*. Once this can be done, the bootstrap distribution of Z can be approximated by taking many bootstrap samples (samples from F^*) as described in the preceding subsection.

A sample, $(\boldsymbol{x}^*_i, y^*_i)$, $i = 1, \ldots, n$, from F^* can be obtained as follows. Generate a new sample, $(\boldsymbol{x}^*_1, \ldots, \boldsymbol{x}^*_n)$, from the same distribution, $F_{\boldsymbol{x}}$, that generated the original sample of inputs, $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. It is assumed here that $F_{\boldsymbol{x}}$ is known. If $F_{\boldsymbol{x}}$ is not known, then $(\boldsymbol{x}^*_1, \ldots, \boldsymbol{x}^*_n)$ can be simply set equal to the original sample $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ or $(\boldsymbol{x}^*_1, \ldots, \boldsymbol{x}^*_n)$ can be produced by sampling with replacement from $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ (i.e. sampling from the empirical CDF of $\boldsymbol{x}$). The corresponding y values, $(y^*_1, \ldots, y^*_n)$, are obtained by

$$y^*_i = \hat{f}(\boldsymbol{x}^*_i) + \varepsilon^*_i, \tag{2.23}$$

where the ε^*_i are sampled with replacement from the model residuals $\varepsilon_i = y_i - \hat{f}(\boldsymbol{x}_i)$, $i = 1, \ldots, n$. As a reminder $\hat{f}(\boldsymbol{x}^*_i)$ is the meta-model constructed from the original sample, $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$, but evaluated at the new sampled values of $\boldsymbol{x}^*_i$. We now have a sample $(\boldsymbol{x}^*_i, y^*_i)$, $i = 1, \ldots, n$ from F^*. This sampling strategy assumes a homogeneous distribution of the errors across the important input variables. If this is not a reasonable assumption, the wild bootstrap could also be used; see p. 247-249 of [21] for more details.

Now repeat this process of obtaining a bootstrap sample as described above M times, so that we have M independent samples of size n from F^*. We denote these M samples as $\{(\boldsymbol{x}^*_1, y^*_1)_k, \ldots, (\boldsymbol{x}^*_n, y^*_n)_k\}$, $k = 1, \ldots, M$. Use each sample, $\{(\boldsymbol{x}^*_1, y^*_1)_k, \ldots, (\boldsymbol{x}^*_n, y^*_n)_k\}$, to construct a new meta-model, $\hat{f}^*_k$. Then, use each $\hat{f}^*_k$ to perform the calculation of T_j as in Eq. (2.1) and denote the result by $\hat{T}^*_{j,k}$, $k = 1, \ldots, M$. Let

$$Z^*_k = \hat{T}^*_{j,k} - \hat{T}_j, \tag{2.24}$$

where $\hat{T}_j$ is the estimate of T_j obtained from the initial meta-model $\hat{f}$. Find the $\alpha/2$ and $(1 - \alpha/2)$ sample quantiles ($z^*_{\alpha/2}$ and $z^*_{1-\alpha/2}$, respectively) from the collection $\{Z^*_1, \ldots, Z^*_M\}$. The CI is then given by Eq. (2.22). A technical detail is that the

lower (or upper) limit of the CI can be less than zero (or greater than one). When this happens the endpoint of the CI can simply be truncated so that the CI is a subset of $[0, 1]$. However, we recommend shifting the interval to be a subset of $[0, 1]$ while maintaining its original length. Our experience suggests that the empirical performance is better (coverages closer to the nominal value) when the interval is shifted instead of truncated.

Bootstrap p-values can also be calculated for the null hypothesis $T_j = 0$. This is accomplished by calculating a lower confidence bound (LCB) for T_j instead of a CI. A LCB for T_j can be obtained by calculating the $(1-\alpha)$ quantile $z^*_{1-\alpha}$. The $100(1-\alpha)\%$ LCB for T_j is then given by

$$L_\alpha = \hat{T}_j - z^*_{1-\alpha}. \tag{2.25}$$

A decision rule for an α level of significance test of hypotheses $H_0 : T_j = 0$ vs. $H_a : T_j > 0$ is given by: If $L_\alpha > 0$ then reject H_0, else do not reject H_0 (i.e. reject H_0 if $\hat{T}_j > z^*_{1-\alpha}$). The p-value for this test is defined as the smallest value of α for which H_0 can be rejected. Thus,

$$\text{p-value} = \inf\{\alpha : \hat{T}_j > z^*_{1-\alpha}\} \tag{2.26}$$

is the desired bootstrap p-value.

The basic bootstrap quantity for T_j in Eq. (2.21) could also be scaled by a variance estimate to produce a t bootstrap as in Eq. (2.18). This requires a variance estimate for $\hat{T}_j$ to be calculated for every bootstrap sample. Unfortunately, the more useful meta-models described in Section 3 result in an $\hat{f}$ that is difficult to study analytically. As $\hat{T}_j$ is a complex functional of $\hat{f}$, precise variance estimates of $\hat{T}_j$ are not available in most cases. The asymptotic variance for a similar estimator of T_j is given on p. 1453 of [22]. This estimate is not exactly valid in our case, but it could still be used to form the t bootstrap statistic. This may produce a distribution of Z^* which is closer to that of Z than the basic bootstrap described above; thus, increasing the accuracy of the bootstrap CIs.

In the test cases we have tried, however, this t bootstrap adjustment was not effective at further increasing the accuracy of the bootstrap CIs. The use of this adjustment is also more complicated and takes somewhat longer computationally. Hence, we recommend using the basic bootstrap and do not provide any computational details about the t bootstrap for $\hat{T}_j$. However, the t bootstrap could still be helpful in other problems than those studied here. In addition, it may be possible to work out a more precise variance estimate for T_j based on the particular meta-model used. This could make the increase in accuracy from the t bootstrap over the basic bootstrap more pronounced and thus worth the additional computation. This is a topic for further study.

3 Meta Models

The success of the strategy discussed in Section 2 depends on the performance of the selected meta-model. As indicated, linear regression remains the most popular choice

for model construction because of its simplicity. When the output is approximately linear in the inputs, it should still be used. However, linear regression will often fail to appropriately identify the importance of certain input variables that have nonlinear effects on the analysis results; see Section 4. In these situations, a more flexible model must be used.

There are many choices of multiple predictor nonparametric regression procedures. Storlie and Helton [12] review some of the more traditional nonparametric regression procedures such as locally weighted polynomial regression (LOESS), additive models (GAMs), projection pursuit regression (PPR), and recursive partitioning (RPART). An implementation of quadratic response surface regression (QREG) was also described. These techniques were shown to be quite useful for SA.

There are several other more state of the art methods that are known to be effective for modeling complex behavior including: Multivariate Adaptive Regression Splines (MARS) [23], Random Forest (RF) [24], Gradient Boosting Machine (GBM) [25], and Adaptive COmponent Selection and Smoothing Operator (ACOSSO) [26]. Gaussian Process (GP) models [27], [28], [11] have also become popular meta-models. We give a description of each of these methods below.

For each of the following procedures, it is assumed that we obtain data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ and that the data were produced through the relation

$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i, \quad i = 1, \ldots, n, \tag{3.1}$$

where f is the function we wish to estimate (i.e. the computer model) and ε_i is an error term. As indicated earlier, the deterministic computer model also fits this framework since the actual model is most often observed with a small amount of numerical error.

3.1 Multivariate Adaptive Regression Splines

The MARS procedure was made popular by Friedman [23]. This method is essentially a combination of spline regression, stepwise model fitting, and recursive partitioning. We first describe the MARS procedure for only one input variable, x, and then generalize to the multiple input setting. Thus, the data we observe is $(x_1, y_1), \ldots, (x_n, y_n)$. For the one input setting, we employ a slight abuse of notation by using x_i to denote the i^{th} observed value of a univariate input x. In all other cases, x_j refers to the j^{th} element of the input vector $\boldsymbol{x}$.

Consider a spline function along the input variable x (usually linear splines are used) with knots at all the distinct data points, i.e.

$$g(x) = b_0 + \sum_{k=1}^{n+1} b_k \phi_k(x), \tag{3.2}$$

where $\phi_1(x) = x$, $\phi_k(x) = |x - x_{k-1}|_+$, $k = 2, \ldots, n+1$, $|x|_+ = x$ if $x > 0$ and 0 otherwise, and $\{b_0, b_1, \ldots, b_{n+1}\}$ are constants. The MARS procedure uses functions of the

form given in Eq. (3.2) to approximate the unknown output function f. Specifically, MARS fits a curve by adding basis functions to a model in a stepwise manner. It starts with a mean only model, then adds the overall linear trend, ϕ_1. From here, the remaining basis functions are added to the model successively in a stepwise manner. Specifically, MARS first fits a model with only the intercept term, i.e.

$$\hat{f}_1(x) = \hat{b}_0, \tag{3.3}$$

where $\hat{b}_0 = \bar{y}$. Call this model 1. Then MARS fits the linear regression model

$$\hat{f}_2(x) = \hat{b}_0 + \hat{b}_1\phi_1(x) \tag{3.4}$$

via least squares and call this model 2. Now MARS adds to model 2 the basis function, ϕ_j, that results in the largest decrease in the Sum of Squares Error (SSE). That is, all models of the form

$$\hat{f}_{3,k}(x) = \hat{b}_0 + \hat{b}_1\phi_1(x) + \hat{b}_k\phi_k(x) \tag{3.5}$$

for $k = 2, \ldots, n+1$ are fit via least squares and the model that reduces $SSE = \sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$ the most is retained. Denote this model 3.

Now a fourth basis function is chosen to be added to model 3 to minimize the SSE. This process is repeated until M basis functions (including the intercept) have been added to the model, where $M \leq n$ is a user defined parameter. The exact value of M will not have much effect on the final estimate as long as it is chosen large enough. In this presentation, M is set to $\min\{n, 200\}$.

At this point, the MARS procedure considers stepwise deletion of basis functions. That is, the current model, f_M, has an intercept term, a linear trend term and $M-2$ other basis functions. Now consider the possible removal of each one of these $M-2$ basis functions. The basis function whose removal will result in the smallest increase in SSE is chosen to be deleted from the model. Typically, the intercept and linear trend, ϕ_1, are not considered candidates for removal. This process is continued until all basis functions have been deleted from the model and only the linear regression model Eq. (3.4) remains.

When this entire process is completed $2M-2$ models have been built (one at each stage) during this stepwise procedure. Each of these models is a candidate for the final model. To choose the "best" model, the generalized cross validation (GCV) score is calculated for each candidate model. Let GCV_l denote the GCV score for the l^{th} model in the stepwise construction, $l = 1, \ldots, 2M-2$. This quantity is defined as

$$GCV_l = SSE_l/(1 - (\nu m_l + 1)/n), \tag{3.6}$$

where m_l and SSE_l are the number of basis functions and the SSE for the l^{th} model, respectively and ν is a penalty (or cost) per basis function. The user defined parameter, ν, essentially acts as a smoothing parameter. That is, smaller values of ν result in smaller models (i.e. fewer basis functions), while larger values of ν result in larger

models. Hence, ν controls the trade off between model complexity and fidelity to the data. In practice, ν is usually chosen between 2 and 4. The model (or set of basis functions) out of the $2M-2$ candidates with the smallest GCV score is chosen as the MARS estimate.

If there is more than one predictor variable, then we switch back to the notation that x_j refers to the j^{th} element of the vector of inputs, $\boldsymbol{x}$. The data we observe is $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$. Consider linear spline functions for each of the input variables x_j with knots at all the distinct data points, i.e.

$$g_j(x_j) = \sum_{l=0}^{n+1} b_{j,l}\phi_{j,l}(x_j), \tag{3.7}$$

where (i) $\{\phi_{j,0} = 1, \phi_{j,1}, \ldots, \phi_{j,n+1}\}$ is a basis for a linear spline on x_j with knots at n distinct data points and (ii) $\{b_{j,0}, b_{j,1}, \ldots, b_{j,n+1}\}$ are constants, $j = 1, \ldots, p$.

For purposes of illustration, assume for the moment that there are $p = 2$ inputs. We wish to extend the linear spline to two dimensions by constructing a model which allows the coefficients of the linear spline on x_1, $\{b_{1,0}, b_{1,1}, \ldots, b_{1,n+1}\}$, to vary as a function of x_2. Specifically, let $b_{1,l}(x_2) = \sum_{m=0}^{n+1} c_{l,m}\phi_{2,l}(x_2)$, $l = 1, \ldots, n+1$, so that each coefficient for the linear spline on x_1 is a linear spline on x_2. Then let,

$$\begin{aligned}
g(\boldsymbol{x}) &= \sum_{l=0}^{n+1} b_{1,l}(x_2)\phi_{1,l}(x_1) \\
&= \sum_{l=0}^{n+1}\sum_{m=0}^{n+1} c_{l,m}\phi_{2,m}(x_2)\phi_{1,l}(x_1) \\
&= [c_{0,0}]_1 + \left[\sum_{l=1}^{n+1} c_{l,0}\phi_{1,l}(x_1) + \sum_{l=1}^{n+1} c_{0,l}\phi_{2,l}(x_2)\right]_2 + \left[\sum_{l=1}^{n+1}\sum_{m=1}^{n+1} c_{l,m}\phi_{1,l}(x_1)\phi_{2,m}(x_2)\right]_3 \\
&= [\text{constant}]_1 + [\text{main effects}]_2 + [\text{two-way interaction}]_3 \,.
\end{aligned} \tag{3.8}$$

The function $g(\boldsymbol{x})$ is called the tensor product spline, since functions of this form can also be constructed as the tensor product of two univariate spline spaces [23]. Notice that the first term in Eq. (3.8) is the intercept, while the next summation includes only functions of one variable (main effects), while the last summation includes functions of two variables (two-way interactions).

For general p, we can write the tensor product spline as

$$\begin{aligned} g(\boldsymbol{x}) &= \sum_{l_1=0}^{n+1}\sum_{l_2=0}^{n+1}\cdots\sum_{l_p=0}^{n+1} c_{l_1,l_2,\ldots,l_p}\phi_{1,l_1}(x_1)\phi_{2,l_2}(x_2)\cdots\phi_{p,l_p}(x_p) \\ &= [d_0]_1 + \left[\sum_{j=1}^{p}\left(\sum_{l=1}^{n+1} d_{j,l}\phi_{j,l}(x_j)\right)\right]_2 + \left[\sum_{j=1}^{p}\sum_{k>j}^{p}\left(\sum_{l=1}^{n+1}\sum_{m=1}^{n+1} d_{j,k,l,m}\phi_{j,l}(x_j)\phi_{k,m}(x_k)\right)\right]_3 \\ &\quad \left[+\cdots+\sum_{l_1=1}^{n+1}\sum_{l_2=1}^{n+1}\cdots\sum_{l_p=1}^{n+1} c_{l_1,l_2,\ldots,l_p}\phi_{1,l_1}(x_1)\phi_{2,l_2}(x_2)\cdots\phi_{p,l_p}(x_p)\right]_4 \\ &= [\text{constant}]_1 + [\text{main effects}]_2 + [\text{2-way interactions}]_3 + [\text{higher interactions}]_4 \end{aligned} \quad (3.9)$$

where the d_0, $d_{j,l}$, etc. correspond to particular values of the $c_{l_1,l_2,\ldots,l_p}$. MARS uses functions of the form given in Eq. (3.9) to approximate the unknown output function f. In most instances, it is not necessary to include all terms from the full tensor product. Often, an additive model (intercept + main effects) will sufficiently model the data. That is, we could consider approximations for f of the form

$$g_{\text{add}}(\boldsymbol{x}) = d_0 + \sum_{j=1}^{p}\left(\sum_{l=1}^{n+1} d_{j,l}\phi_{j,l}(x_j)\right). \quad (3.10)$$

More generally, we could consider the following two-way interaction spline model as an approximation to f:

$$g_{\text{int}}(\boldsymbol{x}) = d_0 + \sum_{j=1}^{p}\left(\sum_{l=1}^{n+1} d_{j,l}\phi_{j,l}(x_j)\right) + \sum_{j=1}^{p}\sum_{k>j}^{p}\left(\sum_{l=1}^{n+1}\sum_{m=1}^{n+1} d_{j,k,l,m}\phi_{j,l}(x_j)\phi_{k,m}(x_k)\right). \quad (3.11)$$

In either case (additive model or two-way interaction model), we can write the approximation as

$$h(\boldsymbol{x}) = d_0 + \sum_{l=1}^{K} d_l\theta_l(\boldsymbol{x}) \quad (3.12)$$

for constants d_l, $l = 1,\ldots,K$, and appropriately defined basis functions θ_l, where $K = p(n+1)$ for the additive model in Eq. (3.10) and $K = p(n+1) + \binom{p}{2}(n+1)^2$ for the two way interaction model in Eq. (3.11). Three way and higher order interaction models can also be considered, but this is not common practice.

The order of the interaction desired for the resulting model must be specified. Once this is done, the MARS algorithm proceeds exactly as in the one input variable case, but with the representation in Eq. (3.12) in place of the representation in Eq. (3.2). That is, MARS first fits the intercept only model,

$$\hat{f}_1(x) = \hat{d}_0, \quad (3.13)$$

where $\hat{d}_0 = \bar{y}$. Then MARS fits all K possible models with two basis functions:

$$\hat{f}_{2,k}(\boldsymbol{x}) = d_0 + d_k\theta_k(\boldsymbol{x}) \tag{3.14}$$

for $k = 1, \ldots, K$ via least squares. The basis function θ_k (from the K possible) that gives the smallest SSE is chosen to be the one that enters the model. Once this basis function is included, MARS looks for the next basis function to add and so on. Once M basis functions have been added, MARS starts to remove basis functions in the same manner as for one input variable. In the end, there are $2M + 1$ possible models and the one with the lowest GCV score is chosen as the MARS estimate.

As a side effect of this model construction, it is common that the final model will not contain any basis functions corresponding to certain input variables. For instance, the final model may have none of the $\phi_{j,l}$ included, effectively removing x_j from the model. Thus, MARS does automatic variable selection (meaning it automatically includes important variables in the model and excludes unimportant ones).

Most implementations of MARS also enforce the restriction that interaction basis functions can enter the model only after both corresponding main effect basis functions are in the model (i.e. $\phi_{j,l}\phi_{k,m}$ is not allowed as a candidate to enter the model until both $\phi_{j,l}$ and $\phi_{k,m}$ have been included). In addition, it is usually the case that the overall linear trend term $\phi_{j,1} = x_j$ must enter the model before any other $\phi_{j,l}$, $l > 1$, terms can enter. In the backwards deletion steps, the $\phi_{j,1}$ is also not allowed to be deleted unless all other $\phi_{j,l}$, $l > 1$, are out of the model first. The implementation used here enforces both of the above restrictions, which increases efficiency and typically results in better variable selection.

3.2 Random Forest

Random Forests (RFs) [24] are machine learning algorithms that were developed originally as classifiers (i.e. for a response whose values are different categories or classes). A RF classifier builds several classification trees by randomly choosing a subset of input variables in a manner to be described below. For a given $\boldsymbol{x}$ value, the output from the RF is the class that is the mode of the classes output by individual trees at that $\boldsymbol{x}$ value. This concept is easily extended to regression by using regression trees in place of classification trees and letting the output of the RF at $\boldsymbol{x}$ be the average value of the output by individual regression trees at $\boldsymbol{x}$.

Regression trees [29] adapt to the input space by splitting the data into disjoint regions (also called nodes). A constant (mean of the data in that region) is then used as the approximation within each region. Each split is made by dividing the observations up at a cutpoint on one variable. Each split is made in a so called greedy fashion. That is, for the first split, a variable and cutpoint along that variable are searched for as to make for the largest increase in R^2 over the mean only model. Then each of these two created regions can be examined for the split which will again make the largest increase in R^2. This process continues until there are J regions (or nodes). See page 267-269 of [29] for a more detailed description of regression trees.

Specifically, the RF is constructed as detailed by *Algorithm 1* below. First, let the number of observations be n, and the number of predictor variables be p. Set the parameter m, where $m < p$ is the number of input variables to be used at each node of the tree. Also set the parameter N_t to be the total number of trees to grow. Lastly, set the parameter n_r, which defines the minimum number of observations to be allowed in each region (or node). By default in the implementation used here, $m = \lfloor p/3 \rfloor$ and $N_t = 500$. The algorithm then proceeds as follows:

Algorithm 1: Construction of Random Forest (RF) Models.

- *For $k = 1, \ldots, N_t$, do*
  1. *Select a training set for this iteration by selecting n observations with replacement from the original n observations (i.e. take a bootstrap sample).*
  2. *Randomly choose m of the p variables. Using these m variables and the training set obtained in step 1, calculate the best variable and cutpoint along that variable to split the data on as in a traditional regression tree. This creates two nodes. The cutpoint is restricted so that each of these two nodes has greater than n_r observations.*
  3. *For each of the two (parent) nodes created in step 2, randomly choose a new m out of the p variables, but continue using the training set obtained in step 1. Using only the m variables chosen in each node respectively, calculate the best variable and cutpoint along that variable to split the data on as in a traditional regression tree. This creates two more nodes for a total of four nodes. Again, the cutpoints are restricted so that each of these four nodes has greater than n_r observations. If either of the parent nodes had less than $2n_r$ observations to begin this step then no split is made on that node.*
  4. *For each of the (parent) nodes created in the previous step, randomly choose a new m out of the p variables, but continue using the training set obtained in step 1. Again split the data in each parent node as in the previous steps unless the parent node has fewer than $2n_r$ observations.*
  5. *Repeat step 4 on all of the (parent) nodes from the previous step until each node has less than $2n_r$ observations.*
  6. *After step 5 we have constructed a regression tree with many nodes, each node with less than $2n_r$ observations. Let this regression tree be denoted $\hat{f}_k$.*
- *The RF estimate is given as*

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N_t} \sum_{k=1}^{N_t} \hat{f}_k(\boldsymbol{x}). \tag{3.15}$$

Random Forest is known to be an accurate meta-model in many cases. It can also handle a large number of input variables and deal with missing data. However, there is no variable selection performed. Because of the large number of trees involved, the final model generally has all variables affecting predictions at least to some extent.

3.3 Gradient Boosting Regression

Like RF, boosting was originally developed for classification purposes [30], [31]. The underlying idea was to combine the output from many "weak" classifiers into a more powerful committee. This has a similar flavor to RF but the concept is actually quite different. Although boosting is applicable to a wide variety of estimation problems, we restrict our attention here to the boosting tree, which estimates a real-valued function. This is a special case of the Gradient Boosting Machine (GBM) framework described in [25].

The general idea behind boosting trees is to compute a sequence of simple trees, where each successive tree is built for the prediction of the residuals from the preceding tree. These trees are then put together in an additive expansion to produce the final estimator.

For a given GBM, each constituent regression tree is restricted to have only J terminal nodes (regions). This distinguishes it from RF where each tree is grown to the point where each node has only a few ($\sim n_r$) observations. The trees involved in the GBM are all relatively simple. For example, in many applications $J = 2$ or 3 would be sufficient This would result in an additive model or a two-way interaction model, respectively, as splits would be allowed on at most one or two variables for each tree in the expansion. However, Friedman suggests that there is seldom much improvement over using $J = 6$ in practice (p. 324 of [29]). That being the case, it is reasonable to use $J = 6$ to allow for more complex interactions should they exist. There is also an N_t parameter corresponding to the number of trees in the expansion. This can be considered a tuning parameter in the sense that R^2 increases as N_t increases. By increasing N_t, we can make the residuals arbitrarily small. The value of N_t can be chosen via cross validation for example. The specific algorithm to fit the boosting tree is as follows:

Algorithm 2: Construction of Gradient Boosting Machine (GBM) Models.

1. *Fit a regression tree with J nodes to the original data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$. That is, search the data for the best variable and cutpoint along that variable to split the data. Repeat this process on each of the two resulting subsets of the data (nodes) to find the best variable and cutpoint to make a split in only one of the regions. Continue until there are J nodes. Call this estimate $\hat{f}_1$.*

2. *For $k = 2, \ldots, N_t$, do*
   (a) *Fit a regression tree with J nodes to the data set $\{\boldsymbol{x}_i, e_{k-1,i}\}_{i=1}^n$ where $e_{k-1,i} = y_i - \sum_{l=1}^{k-1} \hat{f}_{k-1}(\boldsymbol{x}_i)$ are the residuals of the model fit form the previous iteration.*
   (b) *Call this estimate $\hat{f}_k$.*

3. *The final estimate is given as*

$$\hat{f}(\boldsymbol{x}) = \sum_{k=1}^{N_t} \hat{f}_k(\boldsymbol{x}). \tag{3.16}$$

As discussed on p. 326 of [29], the performance of the GBM can be improved by adding a regularization or penalty term to the additive expansion. In step 2 of Algorithm 2, the residuals would then be calculated as $e_i = y_i - \sum_{l=1}^{k-1} \nu \hat{f}_{k-1}(\boldsymbol{x}_i)$, and

$$\hat{f}(\boldsymbol{x}) = \nu \sum_{k=1}^{N_t} \hat{f}_k(\boldsymbol{x}) \tag{3.17}$$

would be used as the final estimate in step 3, where $\nu < 1$ is controlling the "learning rate" of the boosting tree. In [29], it is suggested that $\nu < 0.1$ gives reasonable performance in general. This requires N_t to be larger than with $\nu = 1$, which adds computing time but this strategy generally results in better estimation. We set $\nu = 0.01$ in the implementation used here. Like RF, GBM also works well for a large number of input variables but do not perform variable selection.

3.4 Adaptive COmponent Selection and Shrinkage Operator

The ACOSSO estimate [26] builds on smoothing spline ANOVA (SS-ANOVA) models. To facilitate the description of ACOSSO, we first review the univariate smoothing spline, and then describe tensor product splines which underlie SS-ANOVA framework.

Univariate Smoothing Splines

Again with some abuse of notation, let x_i, $i = 1, \ldots, n$, denote the i^{th} observation of a univariate input x for the discussion of univariate smoothing splines only. Without loss of generality, we restrict attention to the domain $[0, 1]$. We can always rescale the input x to this domain via a transformation. Assume that the unknown function f to be estimated belongs to 2^{nd} order Sobolev space $\mathcal{S}^2 = \{g : g, g' \text{ are absolutely continuous and } g'' \in \mathcal{L}^2[0, 1]\}$. The smoothing spline estimate is given by the minimizer of

$$\frac{1}{n} \sum_{i=1}^{n} \left(y_i - g(x_i)\right)^2 + \lambda \int_0^1 \left[g''(x)\right]^2 dx \tag{3.18}$$

over $g \in \mathcal{S}^2$. The penalty term on the right of (3.18) is an overall measure of the magnitude of the curvature (roughness) of the function over the domain. Thus, the tuning parameter λ controls the trade-off in the resulting estimate between smoothness and fidelity to the data; large values of λ will result in smoother functions while smaller values of λ result in rougher functions but with better agreement to the data. Generally, λ is chosen by generalized cross validation (GCV) ([32]), m-fold CV ([33]), or related methods (p. ?? [34], p. ?? [35]). The minimizer of Eq. (3.18) is technically called the cubic smoothing spline because the solution can be shown to be a natural cubic spline with knots at all of the distinct values of x_i, $i = 1, \ldots, n$ p. ?? [34].

Multidimensional Smoothing Splines

The simplest extension of smoothing splines to multiple inputs is the additive model [35]. For instance, assume that

$$f \in \mathcal{F}_{add} = \{g : g(\boldsymbol{x}) = \sum_{j=1}^{p} g_j(x_j), g_j \in \mathcal{S}^2\}, \tag{3.19}$$

so that f is a sum of univariate functions. The additive smoothing spline is the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(\boldsymbol{x}_i)\right)^2 + \sum_{j=1}^{p}\lambda_j \int_0^1 \left[g_j''(x_j)\right]^2 dx_j \tag{3.20}$$

over $g \in \mathcal{F}_{add}$. The minimizer of the expression in Eq. (3.20), $\hat{f}(\boldsymbol{x}) = \sum_{j=1}^{p}\hat{f}_j(x_j)$, takes the form of a natural cubic spline for each of the functional components $\hat{f}_j$. Notice that there are p tuning parameters for the additive smoothing spline. These are generally determined one at a time by minimizing GCV score with respect to λ_j with the remaining $p-1$ λ_k, $k \neq j$ fixed, and then iterating until convergence as in [35].

To generalize the additive model to allow for two way interactions, we will assume f belongs to the space

$$\mathcal{F}_{2way} = \{g : g(\boldsymbol{x}) = \sum_{j=1}^{p}\sum_{k=j+1}^{p} g_{j,k}(x_j, x_k) : g_{j,k} \in \mathcal{S}^2 \otimes \mathcal{S}^2\}, \tag{3.21}$$

where $\otimes$ represents the tensor product (p. 30-31 of [36]). For two function spaces $\mathcal{G}$ and $\mathcal{H}$, the tensor product space is the vector space generated by (i.e. spanning all linear combinations of) functions of the form gh for $g \in \mathcal{G}$ and $h \in \mathcal{H}$, i.e.

$$\mathcal{G} \otimes \mathcal{H} = \left\{\sum_{k=1}^{N} g_k h_k : g_k \in \mathcal{G}, h_k \in \mathcal{H}, k = 1, \ldots, N\right\}. \tag{3.22}$$

For a complete treatment of tensor product splines and SS-ANOVA, see [37], [38], [39].

We will also need some additional notation to completely specify all of the functional components (main effects and two-way interactions). Let

$$\bar{\mathcal{S}^2} = \{g \in \mathcal{S}^2 : \int_0^1 g(x)dx = 0\} \tag{3.23}$$

and

$$\overline{\mathcal{S}^2 \otimes \mathcal{S}^2} = \{g \in \mathcal{S}^2 \otimes \mathcal{S}^2 : \int_0^1 g(x_1, x_2)dx_1 = \int_0^1 g(x_1, x_2)dx_2 = 0\}. \tag{3.24}$$

Now any function $g \in \mathcal{F}_{2way}$ can be written

$$g = b_0 + \sum_{j=1}^{p} g_j(x_j) + \sum_{k=1}^{p} \sum_{l=k+1}^{p} g_{k,l}(x_k, x_l), \tag{3.25}$$

where b_0 is a constant, $g_j \in \bar{\mathcal{S}}^2$ and $g_{k,l} \in \overline{\mathcal{S}^2 \otimes \mathcal{S}^2}$. The representation in Eq. (3.27) is the functional ANOVA decomposition of g. Notice here that $b_0 = \int g(\boldsymbol{x})d\boldsymbol{x}$ can be interpreted as the overall "average" value of the function. However, technically $b_0 = \mathrm{E}g(\boldsymbol{x})$ only when $\boldsymbol{x}$ has a uniform distribution over $[0,1]^p$. Also, since $\int_0^1 g_{k,l}(x_k, x_l)dx_k = \int_0^1 g_{k,l}(x_k, x_l)dx_l = 0$, the function g_j is truly the main effect function for variable x_j in the sense that

$$g_j(x_j) = \int g(\boldsymbol{x})d\boldsymbol{x}_{(-j)} - b_0, \tag{3.26}$$

where $d\boldsymbol{x}_{(-j)} = dx_1, \ldots, dx_{j-1}, dx_{j+1}, \ldots, dx_p$. Additional background on the preceding relationships is given in [37] and [39].

The two-way interaction smoothing spline is given by the minimizer of

$$\begin{aligned} &\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(\boldsymbol{x}_i)\right)^2 + \sum_{j=1}^{p}\lambda_j \int_0^1 \left[\frac{\partial^2}{\partial x_j^2} g_j(x_j)\right]^2 dx_j + \\ &\qquad \sum_{k=1}^{p}\sum_{l=k+1}^{p}\lambda_{k,l}\int_0^1\int_0^1 \left[\frac{\partial^4}{\partial x_k^2 x_l^2} g_{k,l}(x_k, x_l)\right]^2 dx_k dx_l \end{aligned} \tag{3.27}$$

over $g \in \mathcal{F}_{2way}$. This penalizes the main effect functions exactly the same as before, and also penalizes the two-way interaction functions by a measure of roughness based on a mixed 4^{th} derivative. The minimizer of the expression in Eq. (3.27) can be obtained via matrix algebra using results from reproducing kernel Hilbert space (RKHS) theory; for details see [37], [39]. Notice that this is slightly different from the penalty for the thin plate spline ([37], [38]), which is popular in spatial statistics.

Generalizing to the ACOSSO estimate

The COmponent Selection and Shrinkage Operator (COSSO) [40] penalizes on the sum of the norms instead of the squared norms as in Eqs. (3.20) and (3.27). For ease of presentation, we will restrict attention to the additive model for the remainder of the section. However, all of the following discussion applies directly to the two-way interaction model as well.

The additive COSSO estimate, $\hat{f}(\boldsymbol{x}) = \sum \hat{f}_j(x_j)$, is given by the function $f \in \mathcal{F}_{add}$ that minimizes

$$\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(\boldsymbol{x}_i)\right)^2 + \lambda \sum_{j=1}^{p}\left\{\left[\int_0^1 g_j'(x_j)dx_j\right]^2 + \int_0^1 \left[g_j''(x_j)\right]^2 dx_j\right\}^{1/2}. \tag{3.28}$$

There are three key differences in the penalty term in Eq. (3.28) relative to the additive smoothing spline of Eq. (3.20). First, there is an additional term $\left[\int_0^1 g_j'(x_j)dx_j\right]^2$, which can also be written $[g_j(1) - g_j(0)]^2$, that penalizes the magnitude of the overall trend of the functional component g_j. Second, in contrast to the squared semi-norm in the additive smoothing spline, each term in the sum in the penalty in Eq. (3.28) can be thought of as a norm over functions $g_j \in \bar{\mathcal{S}}^2$. This has a similar effect to the Least Absolute Selection and Shrinkage Operator (LASSO) [41] for linear models in that it encourages some of the terms in the sum to be exactly zero. These terms are norms over the f_j; when such zeros result $\hat{f}_j$ is set to exactly zero, thus providing automatic model selection. Third, the COSSO penalty only has one tuning parameter, which can be chosen via GCV or similar means. It can be demonstrated analytically that the COSSO penalty with one tuning parameter gives as much flexibility as the smoothing spline penalty with p tuning parameters [40].

Finally, the ACOSSO is a weighted version of the COSSO, where a rescaled norm is used as the penalty for each of the functional components. Specifically, we select as our estimate the function $f \in \mathcal{F}_{add}$ that minimizes

$$\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(\boldsymbol{x}_i)\right)^2 + \lambda\sum_{j=1}^{p} w_j \left\{\left[\int_0^1 g_j'(x_j)dx_j\right]^2 + \int_0^1 \left[g_j''(x_j)\right]^2 dx_j\right\}^{1/2}, \quad (3.29)$$

where the w_j, $0 < w_j \leq \infty$, are weights that can depend on an initial estimate of f which we denote $\tilde{f}$. Our implementation of ACOSSO takes $\tilde{f}$ to be the traditional smoothing spline of Eq. (3.20), which is chosen by the GCV criterion with all $\lambda_j = \lambda$. We then use

$$w_j = \left\{\left[\int_0^1 \tilde{f}_j'(x_j)dx_j\right]^2 + \int_0^1 \left[\tilde{f}_j''(x_j)\right]^2 dx_j\right\}^{-1}. \quad (3.30)$$

This allows for more flexible estimation (less penalty) on the functional components that show more signal in the initial estimate. As shown in [26], this approach results in more favorable asymptotic properties than COSSO.

The minimizer of the expression in Eq. (3.29) is obtained using an iterative algorithm and a RKHS framework similar to that used to find the minimizer of Eqs. (3.20) and (3.27) in [37], [39]. The optimization problem for the two-way interaction model can be posed in a similar way to Eq. (3.29); see [26] for details on this and the computation of the solution. The two-way interaction model is used in the results of Sections 4 and 5. As it is a smoothing type method, ACOSSO works best when the underlying function is somewhat smooth. Like the previous methods, ACOSSO also works well when there are a large number of input variables.

3.5 Gaussian Process

The Gaussian Process (GP) for use as a meta-model in computer experiments was first proposed by [27]; see [28] and [11] for additional examples of the use of GPs in

conjunction with computer models. A GP is a stochastic process (random function), $Y(\boldsymbol{x})$, over the space $\boldsymbol{x} \in \mathcal{X}$ such that for any finite set of $\boldsymbol{x}$ values, $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$,

$$\boldsymbol{Y} = [Y(\boldsymbol{x}_1), Y(\boldsymbol{x}_2), \ldots, Y(\boldsymbol{x}_k)]' \tag{3.31}$$

has a multivariate normal distribution. Hence, a GP is completely characterized by its mean and covariance functions

$$\mu_Y(\boldsymbol{x}) = \mathrm{E}[Y(\boldsymbol{x})] \tag{3.32}$$

and

$$K_Y(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}(Y(\boldsymbol{x}), Y(\boldsymbol{x}')), \tag{3.33}$$

respectively. Typically, the meta-model is then defined as

$$\hat{f}(\boldsymbol{x}) = \mathrm{E}[Y(\boldsymbol{x}) \mid Y(\boldsymbol{x}_1) = y_1, Y(\boldsymbol{x}_2) = y_2, \ldots, Y(\boldsymbol{x}_n) = y_n], \tag{3.34}$$

which is the mean of $Y(\boldsymbol{x})$ given the observed values $(\boldsymbol{x}_j, y_j)$, $j = 1, \ldots, n$. The process of obtaining $\hat{f}$ is often called Kriging after Daniel Gerhardus Krige [42].

Since $Y(\boldsymbol{x})$ is Gaussian, the expression for $\hat{f}$ can be given explicitly as

$$\hat{f}(\boldsymbol{x}) = \mu_Y(\boldsymbol{x}) + \boldsymbol{\gamma}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}), \tag{3.35}$$

where

$$\boldsymbol{\gamma} = [K_Y(\boldsymbol{x}_1, \boldsymbol{x}), K_Y(\boldsymbol{x}_2, \boldsymbol{x}), \ldots, K_Y(\boldsymbol{x}_n, \boldsymbol{x})], \tag{3.36}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} K_Y(\boldsymbol{x}_1, \boldsymbol{x}_1) & K_Y(\boldsymbol{x}_1, \boldsymbol{x}_2) & \cdots & K_Y(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ K_Y(\boldsymbol{x}_2, \boldsymbol{x}_1) & K_Y(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots & K_Y(\boldsymbol{x}_2, \boldsymbol{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ K_Y(\boldsymbol{x}_n, \boldsymbol{x}_1) & K_Y(\boldsymbol{x}_n, \boldsymbol{x}_2) & \cdots & K_Y(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}, \tag{3.37}$$

$$\boldsymbol{y} = [y_1, y_2, \ldots, y_n]', \tag{3.38}$$

$$\boldsymbol{\mu} = [\mu_Y(\boldsymbol{x}_1), \mu_Y(\boldsymbol{x}_2) \ldots, \mu_Y(\boldsymbol{x}_n)]', \tag{3.39}$$

and the $(\boldsymbol{x}_j, y_j)$, $j = 1, \ldots, n$ are the previously indicated observed values p. 160-161 [43].

It is possible to assume a constant mean GP and let any trend in the output be accounted for as part of the random process. It is more common, however, to assume that the mean function is linear in the individual x_j. That is,

$$\mu_Y(\boldsymbol{x}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p, \tag{3.40}$$

where the β_j are unknown parameters that need to be estimated from the data.

There are many possible covariance structures one can use; see Section 2 of [44] for a discussion. Here we focus on one very popular class of covariances, the powered exponential family ([45], [11]). This form of this covariance is

$$K_Y(\boldsymbol{x}, \boldsymbol{x}') = \tau^2 \exp\left\{ -\sum_{j=1}^{p} \eta_j |x_j - x_j'|^{\rho_j} \right\}, \tag{3.41}$$

where $\tau^2 = \text{Var}(Y(\boldsymbol{x}))$ is the unconditional, constant (i.e. for all $\boldsymbol{x}$) variance of the process. The η_j, $j = 1, \ldots, p$, referred to as the range parameters, control how far correlation extends in each input direction, and serve the same purpose as the smoothing parameters in a smoothing spline model. The power parameters $0 < \rho_j \leq 2$ control the rate at which the correlation between points decays across the domain. These may be estimated but are typically fixed at $\rho_j = 2$ resulting in an infinitely differentiable process. Values of $0 < \rho < 2$ result in a once differentiable process. These two extremes can be somewhat unsettling which led others to consider the Matern family of covariances with which the user can specify the level of differentiability [46]. However, the powered exponential has more intuitive appeal in terms of understanding how distance controls correlation and remains the most commonly used covariance function for the use of GPs in computer models [27], [28], [11], [47].

Often, it is also useful to allow the observations to have an independent and identically distributed (*iid*) error term as in the traditional frequentist regression models. That is, assume

$$Y(\boldsymbol{x}) = Z(\boldsymbol{x}) + \varepsilon \tag{3.42}$$

where $Z(\boldsymbol{x})$ is a GP with mean and covariance function μ_Z and K_Z, respectively, and ε $N(0, \sigma^2)$, independent for all values of $\boldsymbol{x}$ and $Z(\boldsymbol{x})$ independent of ε. Since the noise process ε and the actual process $Z(\boldsymbol{x})$ are assumed independent, the covariance function for $Y(\boldsymbol{x})$ is obtained by adding what is called the "nugget" term to the covariance function K. That is,

$$K_Y(\boldsymbol{x}, \boldsymbol{x}') = K_Z(\boldsymbol{x}, \boldsymbol{x}') + \sigma^2 I_{\{\boldsymbol{x}=\boldsymbol{x}'\}}, \tag{3.43}$$

where σ^2 is the variance of the *iid* errors, $I_{\{\boldsymbol{x}=\boldsymbol{x}'\}}$ is the indicator function which equals 1 if $\boldsymbol{x} = \boldsymbol{x}'$ and 0 otherwise. The $\sigma^2 I_{\{\boldsymbol{x}=\boldsymbol{x}'\}}$ is what is referred to as the nugget term. The term nugget is borrowed from gold mining to describe an independent source of variability much in the way that gold nuggets tend to be randomly scattered within a mine. Estimation of the model parameters β_j, η_j, τ^2, and σ^2 commonly proceeds via maximum likelihood estimation (MLE); see [48], [28], [49] for details.

It may seem a bit unusual at first to treat a deterministic function (like a computer model), $Y(\boldsymbol{x})$, as a random process. However, this is consistent with a Bayesian way of thinking. For example, the GP that we use represents our prior belief of what our computer model output will produce. Before the output is evaluated at the design points, these output values are unknown to us. However, we may have some preconceived notion about what the output will look like. For example, we may believe that the underlying output function is "smooth" in some sense. The GP model represents this subjective uncertainty about what the output might look like. For instance, if we generate several realizations from the GP model unconditionally, we would expect that the computer output would look something like one of these realizations if or when it is produced. Thus, the GP is really our prior distribution on the output.

One benefit to this Bayesian framework is that it produces both an expected value of an output function f given the data and an entire posterior distribution of f given

the observed values. That is for a set $A \subset \Re^k$, one can obtain

$$\Pr([f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_k)] \in A \mid \boldsymbol{Y} = \boldsymbol{y}) \tag{3.44}$$

for any set of points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k$, where $\boldsymbol{Y}$ is the random vector of observed values defined in Eq. (3.31) and $\boldsymbol{y} = [y_1, \ldots, y_n]'$ is a vector of constants. This posterior distribution is multivariate Gaussian and is determined by calculations similar to those underlying in Eq. (3.35) to calculate the conditional mean and variance (see p. 160-161 of [43]). This posterior distribution can be used to develop a Bayes estimate for T_j along with Bayesian credible sets for the T_j in lieu of $\hat{T}_j$ from Eq. (2.1) and the bootstrap approach described in 2.3. A $100(1-\alpha)\%$ Bayesian credible set for a parameter θ is defined to be a any set A for which $\Pr(\theta \in A \mid \boldsymbol{Y} = \boldsymbol{y}) = 1 - \alpha$.

To create a Bayes estimate and credible set for T_j, we need to perform the following steps:

(1) generate two samples of size N from the $\boldsymbol{x}$ distribution in the same manner as used to calculate T_j described on p. 178 of [8].

(2) For each sample in (1) we can generate the vector $[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_N)]$ from the conditional distribution in Eq. (3.44) and carry out the calculation of T_j using these values. This constitutes a sampled value of T_j from the posterior distribution, (i.e. the distribution of T_j given $\boldsymbol{Y} = \boldsymbol{y}$).

(3) Repeat steps (1) and (2) M times to produce M independent draws of T_j, denoted $\{T_{j,1}, \ldots, T_{j,M}\}$ from the posterior distribution. It is assumed that M is reasonably large (e.g. 1000) so that the posterior distribution is well represented by $\{T_{j,1}, \ldots, T_{j,M}\}$. A Bayes estimate for T_j is given by the posterior mean of T_j,

$$\tilde{T}_j = \frac{1}{M} \sum_{m=1}^{M} T_{j,m}. \tag{3.45}$$

A $100(1-\alpha)\%$ credible set is given by

$$(T_{j,\alpha/2}, T_{j,1-\alpha/2}), \tag{3.46}$$

where $T_{j,\alpha/2}$ and $T_{j,1-\alpha/2}$ are the $\alpha/2$ and $1-\alpha/2$ sample quantiles of $\{T_{j,1}, \ldots, T_{j,M}\}$.

Technically we do not have to repeat Step (1) in Step (3) but it is beneficial to do so in order to account for Monte Carlo (i.e. sampling) error in the numerical evaluation of the T_j used in the determination of credible intervals.

If the free parameters of the model ($\beta_j's$, τ, σ^2 ,η_j's) are estimated via MLE or by some other means and then treated as fixed (as is suggested here), then this procedure is referred to as an empirical Bayesian approach. It is well known, however, that this approach can underestimate the variance in the predictions of new observations [50]. Hence, when using this approach to estimate uncertainty or sensitivity indices, we

still recommend using the kriging estimate, $\hat{f}$, to produce the estimates of T_j as in Eq. (2.1) in conjunction with the bootstrap procedure to produce confidence intervals.

Because of the inherent Bayesian nature of the GP approach, it is also becoming common practice to put hyper-priors on the parameters in the mean and covariance functions to make the procedure "fully" Bayesian (i.e. no MLE estimation involved). We do not give details of this here, but refer the reader to [47], [45] and references therein. When the estimation is fully Bayesian, the credible sets discussed in the preceding paragraph become a more natural way to represent the uncertainty in the estimate.

3.6 Discussion of Meta-Model choice

As is to be expected, the performance of the various procedures will vary widely from one application to another. For example, quadratic regression [12] will typically outperform more complex methods when the true surface can be well approximated by a quadratic function. In a similar manner, ACOSSO will outperform MARS and Recursive Partitioning [12] when the true surface is sufficiently smooth. However, recursive partitioning is very well suited for modeling functions when discontinuities are present. In short, all of the methods have advantages and disadvantages. And sometimes disadvantages in one situation become advantages in another.

The one thing that seems to be the most useful in a building a meta-model for a complex computer model with many inputs is some form of variable selection. This becomes very important for both accuracy and computational efficiency when there are a large number of input variables. For example, suppose there are 30 input variables and only 5 of them significantly affect the output. This is a fairly typical situation in the real analyses of the type illustrated in Section 5.2. A method that incorporates variable selection like MARS will have a model estimate with perhaps 7 variables selected. Hence, S_j and T_j for the other 23 variables are 0; so we do not need to calculate them. This eliminates a substantial portion of the computational burden. Methods like GBM and RF work very well for approximating a surface in higher dimensions. However, there is no variable selection inherent to the fit. Hence, when using these methods it becomes necessary to calculate all 30 T_j values. Even though most T_j values will be very close to zero, we will not know which T_j's until we calculate them.

Of course, one can always incorporate variable selection into any procedure via some form of thresholding or a stepwise/stagewise/best subset type model selection approach. However, with complex meta-models, it is not always clear what an appropriate criterion to use for model selection should be. In addition, stepwise type model fitting can be more of a burden when a complex model fitting procedure is involved. This is because it will typically take up much more computing time and negate any computational savings to be gained by the variable selection.

The lack of variable selection is also part of the reason why we do not recommend the use of standard Gaussian process type models as in [11, 51] when there is a

large number of input variables. In addition to the inefficiency of calculating T_j values, there is also a concern about estimation accuracy. A Gaussian process in 30 dimensions, for example, can be hampered by the curse of dimensionality. This is due to the fact that a GP allows for 30 way and all lower order interactions to be present in the resulting model fit. When different smoothness parameters are allowed for each $\boldsymbol{x}$ component direction, this problem is partially alleviated but it may still be an issue in many problems. There are, however, some very recent works pertaining to variable selection for Gaussian processes [52, 47, 53] that might make these methods more practical for our purposes here. This is a topic for further study.

4 Simulation Examples

In this section we investigate the properties of the proposed methodology for estimating T_j and the corresponding CIs on various scenarios where the actual values can be calculated. We will use three example outputs also used in several previous studies [54, 8, 55, 56]. These are:

$$y_1 = f_1(\boldsymbol{x}) = \frac{(x_2 + 0.5)^4}{(x_1 + 0.5)^2} \tag{4.1}$$

$$y_2 = f_2(\boldsymbol{x}) = 2\prod_{j=1}^{10} \frac{|4x_j - 2| + a_j}{1 + a_j} \tag{4.2}$$

$$\text{with } [a_1, a_2, \ldots, a_{10}] = [0, 1, 2, 4, 8, 99, 99, 99, 99, 99]$$

$$y_3 = f_3(\boldsymbol{x}) = \sin(2\pi x_1 - \pi) + 7\sin^2(2\pi x_2 - \pi) + 0.1(2\pi x_3 - \pi)^4 \sin(2\pi x_1 - \pi). \tag{4.3}$$

It is assumed that the output we actually observe is subject to a small amount of error to mimic the numerical error present in a real application, that is we observe $y_j = f_j(\boldsymbol{x}) + \varepsilon_j$. In all the examples, we let the ε_j terms be generated as $iid\, \mathcal{N}(0, 0.25)$ variables. This produces signal to noise ratios (SNRs) of 2760:1, 8:1, and 55:1 for y_1, y_2, and y_3, respectively, where $SNR = \text{Var}(f(\boldsymbol{x}))/Var(\varepsilon)$.

In this example model there are 3 output variables and 10 input variables (x_j's), although not all of the inputs have an affect on each of the three outputs. Such multiple outputs are usually the case in analyses of real systems (i.e., see the analyses in [57, 7] from which the examples in Section 5.2 are derived). Further, it is also typical of such analyses that individual results are not affected by all of the uncertain variables under consideration.

The individual models are functions of anywhere from 2 to 10 input variables. There are several completely uninformative inputs in output models $f_1 1$ and f_3. Thus, these examples will test the ability of the methodology to identify important inputs while also testing their ability to disregard input variables that are uninformative to a particular analysis. The functions f_1 and f_2 and the associated distributional assumptions for the x_j correspond to Models 6b, 7 and 9, respectively, in [54] and Models 2-4 in Section 3 of [56]. The functions f_2 and f_3 are also considered in Sections

4 and 5 of [55]. Although, here f_2 has been adjusted slightly to involve all 10 inputs as opposed to only the first 8 inputs.

These analytic models have an advantage over the real model considered in the next section (Section 5.2). Specifically, they are fast enough to evaluate so that it is possible to calculate with great precision any quantity we wish such as the true values for S_j and T_j of Eq. (1.3) and Eq. (2.5). This is not possible with a computationally demanding model of the type considered in Section 5.2, which is of course why we need to use a meta-model for such calculations. In short, these examples make comparisons between truth and sensitivity results obtained with the procedures under consideration possible.

We consider the following meta-models in this evaluation: linear regression (REG), quadratic regression (QREG), Additive Models (GAMs), Recursive Partitioning (RPART), Multivariate Adaptive Regression Splines (MARS), Adaptive COmponent Selection and Smoothing Operator (ACOSSO), Random Forests (RF), Gradient Boosting Method (GBM), Gaussian Processes (GP) with MLE and Bootstrap CIs (MLE GP), and GP with MLE, but with Bayes estimates and Bayesian Credible Sets for T_j given in Eqs. (3.45) and (3.46) (MLE BGP). Discussions of MARS, ACOSSO, RF, GBM, and GP are given in Sect. 3 of this presentation, and similar discussions of REG, QREG, GAMs, and RPART are given in Ref. [12]. All of these meta-models were constructed using the *CompModSA* R package available at `http://www.stat.unm.edu/ storlie`.

To evaluate the various models, we generate 100 random samples (realizations) each with a sample size of $n = 300$. For each of the 100 samples we evaluate the model at the n sample points to obtain the three outputs. We then construct point estimates and bootstrap confidence interval estimates for the T_j given by each of the above methods for each sample. This allows us to evaluate the "long-run" performance of each approach. We then consider the effect of varying the sample size in Section 4.4

There are several criteria which we will use to test our procedure for the various meta-models. A typical measure to use when comparing several possible estimators is the Root Mean Squared Error (RMSE) of the estimate. The RMSE for an estimate $\hat{T}_j$ is given as $\{\mathrm{E}[(\hat{T}_j - T_j)^2]\}^{1/2}$. In the examples of this section we can calculate the true value T_j, and what we actually report is a Monte Carlo estimate of RMSE by taking the average squared error over the 100 realizations. Let $\hat{T}_{j,k}$ denote the value of $\hat{T}_j$ for the k^{th} realization. Then,

$$R = \sqrt{\frac{1}{100}\sum_{k=1}^{100}(\hat{T}_{j,k} - T_j)^2} \tag{4.4}$$

is the summary measure presented in this section. It is also useful to estimate the standard deviation of R to know how variable this *RMSE* estimate is when comparing its value across meta-models. An estimate of the standard deviation of R can be calculated using a Taylor approximation

$$s_R = \frac{1}{2R} s_{R^2} \tag{4.5}$$

where

$$s^2_{R^2} = \frac{1}{99}\sum_{k=1}^{100}\left[(\hat{T}_{j,k} - T_j)^2 - R^2\right]^2. \tag{4.6}$$

Keep in mind when comparing results that R is not necessarily comparable across differing values of the T_j. For instance, it would usually be considered a bigger error for $\hat{T}_j$ to be off by 0.10 when $T_j = 0.02$ than it would be if $T_j = 0.70$.

We also calculate the coverage and the average length of 95% bootstrap confidence intervals calculated using the bootstrap approach described in Section 2.3. Let the 95% CI for T_j for the k^{th} realization be denoted $(\hat{T}_{j,k,L}, \hat{T}_{j,k,U})$. Then,

$$C = \frac{1}{100}\sum_{k=1}^{100} I_{(\hat{T}_{j,k,L}, \hat{T}_{j,k,U})}(T_j) \tag{4.7}$$

$$L = \frac{1}{100}\sum_{k=1}^{100}(\hat{T}_{j,k,U} - \hat{T}_{j,k,L}) \tag{4.8}$$

define the two indicated summary measures, where $I_A(x)$ is the indicator function defined by $I_A(x) = 1$ if $x \in A$ and 0 otherwise. We present the preceding summary measures of for each input variable for all of the meta-models listed above for each of the 3 outputs defined in Eqs. (4.1) - (4.3).

4.1 Output y_1

The first output is monotonic across each of the inputs. What makes this SA difficult however is the substantial interaction between x_1 and x_2 as can be seen in Figure 1.

Figure 2 summarizes the results for the 100 realizations of y_1 with a sample size of $n = 300$. Each panel is a boxplot of the 100 $\hat{T}_j$, $j = 1, \ldots, 10$, from each realization for the corresponding meta-model. Dashed lines are drawn at the corresponding true T_j values for reference. Here we see that each meta-model does well to estimate the T_j for the unimportant inputs (x_3-x_{10}), with the exception of MLE BGP. However, some methods show considerable biases for $\hat{T}_j$ of the important inputs. For instance, GAM cannot model the important interaction between x_1 and x_2 in this example so it underestimates the total effect for both inputs. TREE has a systematic upward bias for $\hat{T}_2$. RF has a substantial downward bias for $\hat{T}_1$. GBM has a downward bias for $\hat{T}_1$ and an upward bias for $\hat{T}_2$. QREG also has a slight downward bias for $\hat{T}_1$. MARS, ACOSSO, MLE GP, and MLE BGP, all have distributions for $\hat{T}_1$ and $\hat{T}_2$ centered right near the corresponding true values. In addition MARS and ACOSSO have $\hat{T}_j$ distributions for uninformative variables that are concentrated around 0, with small variability. The $\hat{T}_j$ distributions for uninformative variables for MLE GP are concentrated at 0, but with more variability than those for MARS and ACOSSO. Lastly, MLE BGP has a substantial upward bias for the $\hat{T}_j$ of the uninformative variables.

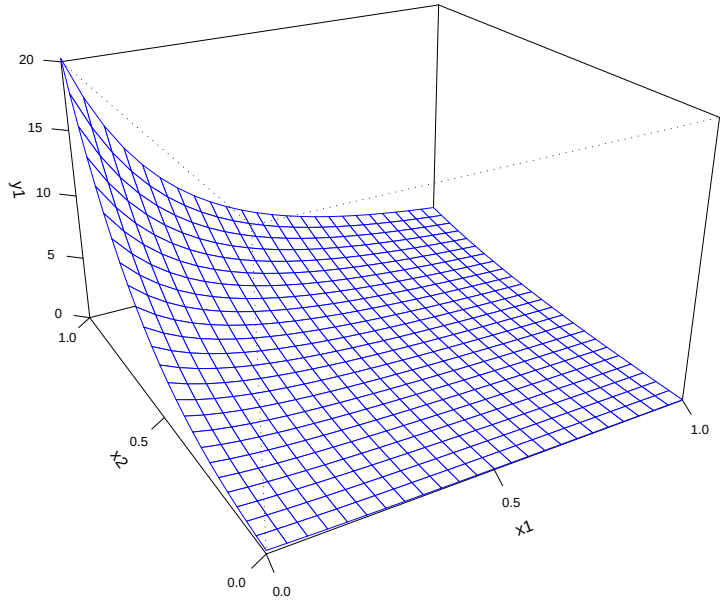


Figure 1: Output y_1 plotted against inputs x_1 and x_2.

Table 1 further summarizes the results of estimating T_j for output y_1. The true T_j for each input are given in the first column for perspective. The last row of the results for each meta-model contains the averages of the summary measures across all of the important input variables. For this presentation, we define an input to be important with respect to a particular output if $T_j > .10$. Thus, the last row gives an overall summary of how well the meta-model performed to estimate the effect of the important inputs. This makes it easier to compare one meta-model to another.

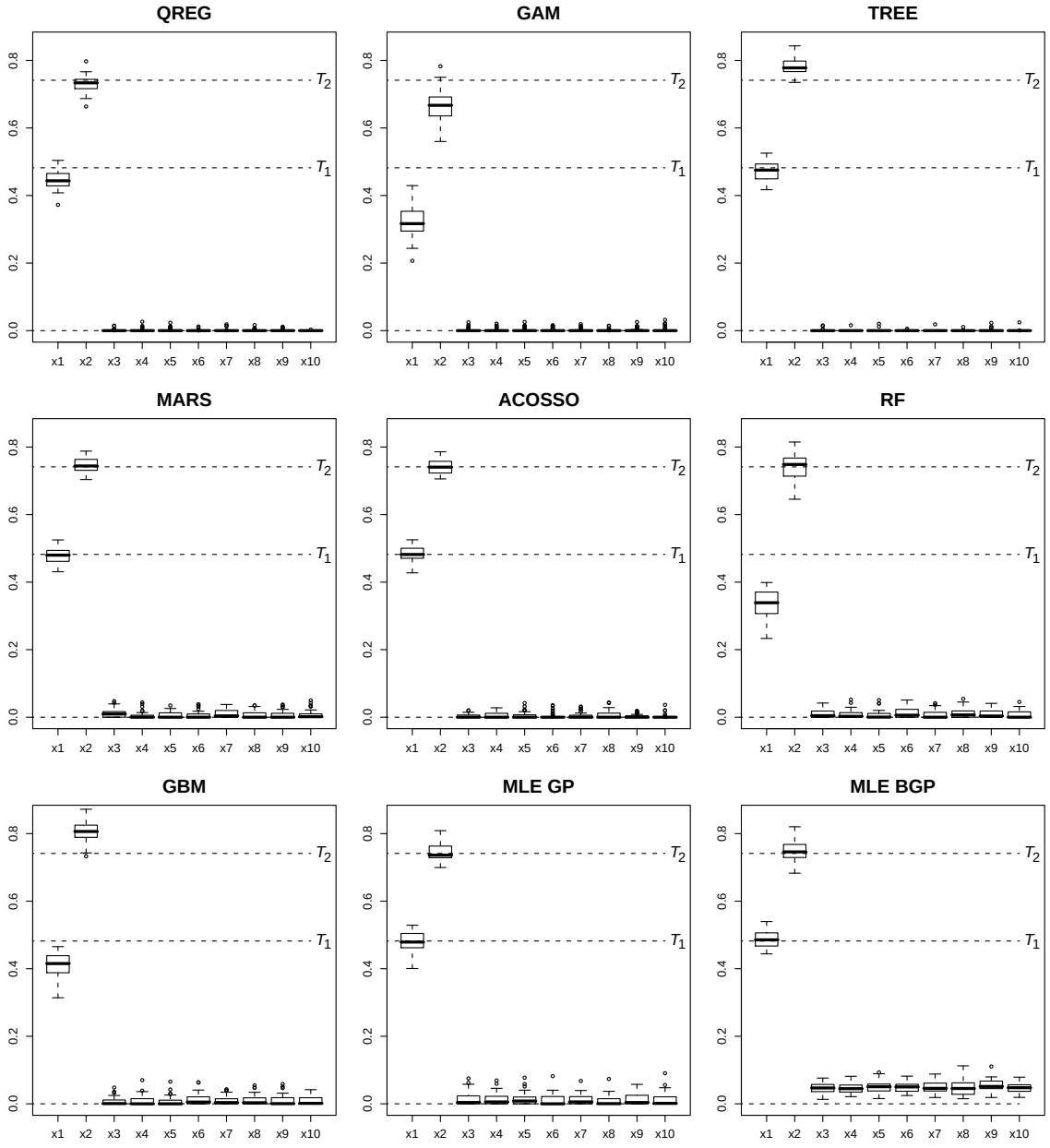


Figure 2: Boxplots of the $\hat{T}_j$ for each of the meta-models for the output y_1. Dashed lines are drawn at the corresponding true values of T_j for reference.

Table 1. Results for y_1 with 100 samples of size $n = 300$.

Var[a]	T_j[b]	RMSE[c]	Cover[d]	Length[e]	Var	T_j	RMSE	Cover	Length
		REG					**QREG**		
x1	0.48	0.17 (0.01)	0.00 (0.00)	0.15 (0.00)	x1	0.48	0.04 (0.00)	0.66 (0.08)	0.10 (0.00)
x2	0.74	0.07 (0.00)	0.80 (0.07)	0.15 (0.00)	x2	0.74	0.03 (0.00)	0.91 (0.05)	0.10 (0.00)
x3	0.00	0.00 (0.00)	0.97 (0.03)	0.02 (0.00)	x3	0.00	0.00 (0.00)	0.97 (0.03)	0.02 (0.00)
x4	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x4	0.00	0.01 (0.00)	0.97 (0.03)	0.02 (0.00)
x5	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x5	0.00	0.00 (0.00)	0.97 (0.03)	0.02 (0.00)
x6	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x6	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
x7	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x7	0.00	0.00 (0.00)	1.00 (0.00)	0.01 (0.00)
x8	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x8	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
x9	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x9	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
x10	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	x10	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
avg.[f]		0.12 (0.03)	0.40 (0.28)	0.15 (0.00)	avg.		0.04 (0.00)	0.79 (0.09)	0.10 (0.00)
		GAM					**RPART**		
x1	0.48	0.16 (0.01)	0.00 (0.00)	0.15 (0.00)	x1	0.48	0.03 (0.00)	0.91 (0.05)	0.11 (0.00)
x2	0.74	0.09 (0.01)	0.06 (0.04)	0.15 (0.00)	x2	0.74	0.05 (0.00)	0.69 (0.08)	0.10 (0.00)
x3	0.00	0.01 (0.00)	1.00 (0.00)	0.04 (0.00)	x3	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x4	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)	x4	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x5	0.00	0.01 (0.00)	0.94 (0.04)	0.03 (0.00)	x5	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x6	0.00	0.00 (0.00)	1.00 (0.00)	0.03 (0.00)	x6	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x7	0.00	0.00 (0.00)	0.94 (0.04)	0.03 (0.00)	x7	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x8	0.00	0.00 (0.00)	0.97 (0.03)	0.02 (0.00)	x8	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x9	0.00	0.00 (0.00)	0.91 (0.05)	0.02 (0.00)	x9	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
x10	0.00	0.01 (0.00)	0.83 (0.06)	0.03 (0.00)	x10	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
avg.		0.13 (0.02)	0.03 (0.02)	0.15 (0.00)	avg.		0.04 (0.01)	0.80 (0.08)	0.11 (0.01)
		MARS					**ACOSSO**		
x1	0.48	0.03 (0.00)	0.97 (0.03)	0.11 (0.00)	x1	0.48	0.02 (0.00)	0.86 (0.06)	0.10 (0.00)
x2	0.74	0.02 (0.00)	0.94 (0.04)	0.11 (0.01)	x2	0.74	0.02 (0.00)	0.91 (0.05)	0.11 (0.00)
x3	0.00	0.02 (0.00)	0.97 (0.03)	0.08 (0.00)	x3	0.00	0.01 (0.00)	1.00 (0.00)	0.03 (0.00)
x4	0.00	0.01 (0.00)	0.94 (0.04)	0.07 (0.00)	x4	0.00	0.01 (0.00)	0.97 (0.03)	0.04 (0.00)
x5	0.00	0.01 (0.00)	0.97 (0.03)	0.07 (0.00)	x5	0.00	0.01 (0.00)	0.97 (0.03)	0.04 (0.00)
x6	0.00	0.01 (0.00)	0.91 (0.05)	0.08 (0.00)	x6	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)
x7	0.00	0.02 (0.00)	1.00 (0.00)	0.08 (0.00)	x7	0.00	0.01 (0.00)	0.97 (0.03)	0.04 (0.00)
x8	0.00	0.01 (0.00)	0.97 (0.03)	0.08 (0.00)	x8	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)
x9	0.00	0.01 (0.00)	0.97 (0.03)	0.08 (0.00)	x9	0.00	0.01 (0.00)	1.00 (0.00)	0.04 (0.00)
x10	0.00	0.02 (0.00)	0.94 (0.04)	0.08 (0.00)	x10	0.00	0.01 (0.00)	0.97 (0.03)	0.04 (0.00)
avg.		0.02 (0.00)	0.96 (0.01)	0.11 (0.00)	avg.		0.02 (0.00)	0.89 (0.02)	0.10 (0.00)
		Random Forest					**GBM**		
x1	0.48	0.15 (0.01)	0.71 (0.08)	0.17 (0.00)	x1	0.48	0.08 (0.01)	1.00 (0.00)	0.26 (0.01)
x2	0.74	0.04 (0.00)	0.94 (0.04)	0.17 (0.00)	x2	0.74	0.07 (0.00)	0.83 (0.06)	0.20 (0.01)
x3	0.00	0.02 (0.00)	0.86 (0.06)	0.06 (0.00)	x3	0.00	0.02 (0.00)	0.97 (0.03)	0.08 (0.00)
x4	0.00	0.02 (0.00)	0.94 (0.04)	0.06 (0.00)	x4	0.00	0.02 (0.00)	0.97 (0.03)	0.08 (0.00)
x5	0.00	0.02 (0.00)	0.91 (0.05)	0.06 (0.00)	x5	0.00	0.01 (0.00)	0.97 (0.03)	0.08 (0.00)
x6	0.00	0.02 (0.00)	0.86 (0.06)	0.06 (0.00)	x6	0.00	0.02 (0.00)	0.91 (0.05)	0.08 (0.00)
x7	0.00	0.02 (0.00)	0.89 (0.05)	0.06 (0.00)	x7	0.00	0.02 (0.00)	0.97 (0.03)	0.09 (0.00)
x8	0.00	0.02 (0.00)	0.94 (0.04)	0.06 (0.00)	x8	0.00	0.02 (0.00)	0.94 (0.04)	0.08 (0.00)
x9	0.00	0.02 (0.00)	0.89 (0.05)	0.06 (0.00)	x9	0.00	0.02 (0.00)	0.94 (0.04)	0.08 (0.00)
x10	0.00	0.02 (0.00)	0.89 (0.05)	0.06 (0.00)	x10	0.00	0.02 (0.00)	1.00 (0.00)	0.09 (0.00)

Table 1. Results for y_1 with 100 samples of size $n = 300$.

Var	T_j	RMSE	Cover	Length	Var	T_j	RMSE	Cover	Length
avg.		0.10 (0.04)	0.83 (0.08)	0.17 (0.00)	avg.		0.07 (0.00)	0.91 (0.06)	0.23 (0.02)
		MLE GP					**MLE BGP**		
x1	0.48	0.03 (0.00)	0.89 (0.05)	0.13 (0.00)	x1	0.48	0.03 (0.00)	1.00 (0.00)	0.20 (0.00)
x2	0.74	0.03 (0.00)	0.77 (0.07)	0.12 (0.00)	x2	0.74	0.03 (0.00)	1.00 (0.00)	0.20 (0.01)
x3	0.00	0.03 (0.00)	0.83 (0.06)	0.08 (0.00)	x3	0.00	0.05 (0.00)	1.00 (0.00)	0.18 (0.01)
x4	0.00	0.02 (0.00)	0.89 (0.05)	0.09 (0.00)	x4	0.00	0.05 (0.00)	1.00 (0.00)	0.18 (0.01)
x5	0.00	0.02 (0.00)	0.94 (0.04)	0.10 (0.00)	x5	0.00	0.06 (0.00)	1.00 (0.00)	0.18 (0.01)
x6	0.00	0.02 (0.00)	0.91 (0.05)	0.09 (0.00)	x6	0.00	0.05 (0.00)	1.00 (0.00)	0.18 (0.01)
x7	0.00	0.02 (0.00)	0.97 (0.03)	0.10 (0.00)	x7	0.00	0.05 (0.00)	1.00 (0.00)	0.17 (0.01)
x8	0.00	0.02 (0.00)	0.94 (0.04)	0.09 (0.00)	x8	0.00	0.05 (0.00)	1.00 (0.00)	0.18 (0.01)
x9	0.00	0.02 (0.00)	0.94 (0.04)	0.10 (0.00)	x9	0.00	0.06 (0.00)	1.00 (0.00)	0.19 (0.01)
x10	0.00	0.03 (0.00)	0.89 (0.05)	0.09 (0.00)	x10	0.00	0.05 (0.00)	1.00 (0.00)	0.19 (0.01)
avg.		0.03 (0.00)	0.83 (0.04)	0.12 (0.00)	avg.		0.03 (0.00)	1.00 (0.00)	0.20 (0.00)

[a] input variable name.

[b] True value of T_j.

[c] Monte Carlo approximation of the RMSE of $\hat{T}_j$ as defined in (4.4) for the corresponding input and meta-model. The estimated standard deviation of the RMSE as defined in (4.6) is given in parantheses.

[d] Coverage as defined in (4.7) of the CI's produced for T_j for the corresponding input and meta-model. The estimated standard deviation of Coverage is given in parantheses.

[e] Length as defined in (4.8) of the CI's produced for T_j for the corresponding input and meta-model. The estimated standard deviation of Length is given in parantheses.

[f] Average value of each column using only the inputs with $T_j \geq 0.1$.

Notice that REG has a large RMSE (0.17) for estimating the total variance of the first input, x_1. The 95% CI for T_1 also has 0% coverage. This is due the inability on the part of linear regression to model the curvature in y_1 across x_1 as well as th interaction effect between x_1 and x_2. All of these examples are nonlinear and we do not expect REG to perform well, but rather to serve as a baseline for how much improvement can be made. QREG on the other hand has a much better RMSE for T_1 and a coverage of (0.66) which is still a bit lower than the nominal level (0.95). GAM also struggles a bit with RMSE and coverage in this example because of GAM's inability to model the important interaction between x_1 and x_2. RPART does about as well as QREG in this example with small RMSE and also has good coverage for the CIs with the exception of that for T_2 (0.69). MARS does very well in terms of RMSE and coverage for the first two inputs. However it has somewhat larger error and coverages lower than 0.95 for the unimportant input variables. MARS seems to be frequently estimating the $T_j = 0$ to be greater than zero in this example.

ACOSSO and MARS are two of the best methods for this example with the smallest RMSE for all of the inputs, coverages close to the nominal level, and also small average CI lengths. ACOSSO has a very good CI coverages in general. The

coverage for ACOSSO T_1 CI is 0.86, while all of the MARS CIs maintain closer to 0.95 coverage. However, the ACOSSO CIs are all somewhat narrower (shorter average length) than those from MARS, especially for the uninformative variables.

Random Forest has some difficulty relative to the other methods on this example. It has RMSE and coverages for T_1 and T_2 much better than REG, but not nearly as good as other methods. GBM seems to perform very similarly to Random Forest but with better coverage on T_1 CIs while the coverage for T_2 is worse, even though the average CI length for T_2 is quite big (0.20).

The results for MLE GP are similar to ACOSSO and MARS in that the RMSE and the average CI lengths are small compared to the other methods . However the coverages for T_1 (0.89) and particularly T_2 (0.77) are not as high as for the other methods. MLE BGP has low RMSE for T_1 and T_2 as well, but slightly higher RMSE for the uninformative variables. The coverages are very high (1.00) for all T_j's but this is not necessarily good since the CI length is large (near 0.20) for all of the CIs.

4.2 Output y_2

The second output is the non-monotonic Sobol g-function. Here, there is substantial nonlinearity and interaction; see Figure 3.

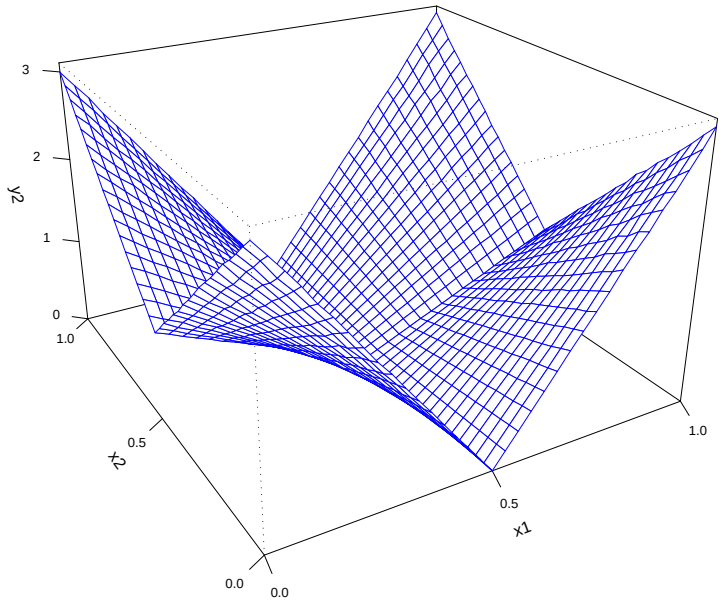


Figure 3: Plot of output y_2 against inputs x_1 and x_2 with other inputs fixed at $x_j = 0.5$ for $j = 3, \ldots, 10$.

Figure 4 displays the boxplots for each meta model of the $\hat{T}_j$ distribution, $j = 1, \ldots, 10$ for the output y_2. Dashed lines are drawn at the corresponding true T_j

values for reference. Here we see that none of the meta-models estimate all of the T_j without bias, although clearly some are better than others. QREG has reasonable performance overall. It has a downward bias for $\hat{T}_1$ and $\hat{T}_2$, but the other $\hat{T}_j$ are nicely centered around the corresponding T_j lines. GAM also has downward bias for $\hat{T}_1$ and $\hat{T}_2$ (likely because of the interactions involved here), as well as a slight bias for $\hat{T}_3$, but is right on with the other $\hat{T}_j$. TREE has a substantial upward bias for $\hat{T}_1$ but is pretty close to T_j with the other $\hat{T}_j$. Notice, however, that TREE has much more variability in the $\hat{T}_j$ for $j > 4$ than many of the other methods (e.g. ACOSSO, RF, GBM, MLE GP). MARS has reasonable performance overall, especially for $\hat{T}_2$, $\hat{T}_3$, and $\hat{T}_4$. However, $\hat{T}_1$ is biased low with substantial variability, and $\hat{T}_j$ for $j > 4$ have more variability than some of the other methods. ACOSSO has good performance in general, $\hat{T}_1$, $\hat{T}_2$, and $\hat{T}_3$ are all slightly downward biased, but are close to the true T_j values with little variability. Also, the $\hat{T}_j$ for $j > 4$ are very tightly centered around the true values. RF has a moderate bias for $\hat{T}_1$ - $\hat{T}_3$, and also has $\hat{T}_j$ for $j > 4$ very tightly centered around the true values. GBM has very substantial biases for $\hat{T}_1$ - $\hat{T}_3$. MLE GP has good performance for $\hat{T}_1$ and $\hat{T}_2$ but the distribution for $\hat{T}_3$ is highly variable and the $\hat{T}_4$ is also biased low. MLE BGP has a large bias problem for all $\hat{T}_j$.

Additional detail of the results for y_2 is summarized in Table 2. REG really has trouble with this example as it is highly nonlinear. However, QREG does very well in terms of RMSE. The coverage for T_1 and T_2 is only 0.73 and 0.76, respectively, but the coverages for the other inputs are near the nominal level. This is reasonable especially when compared to the rest of the methods.

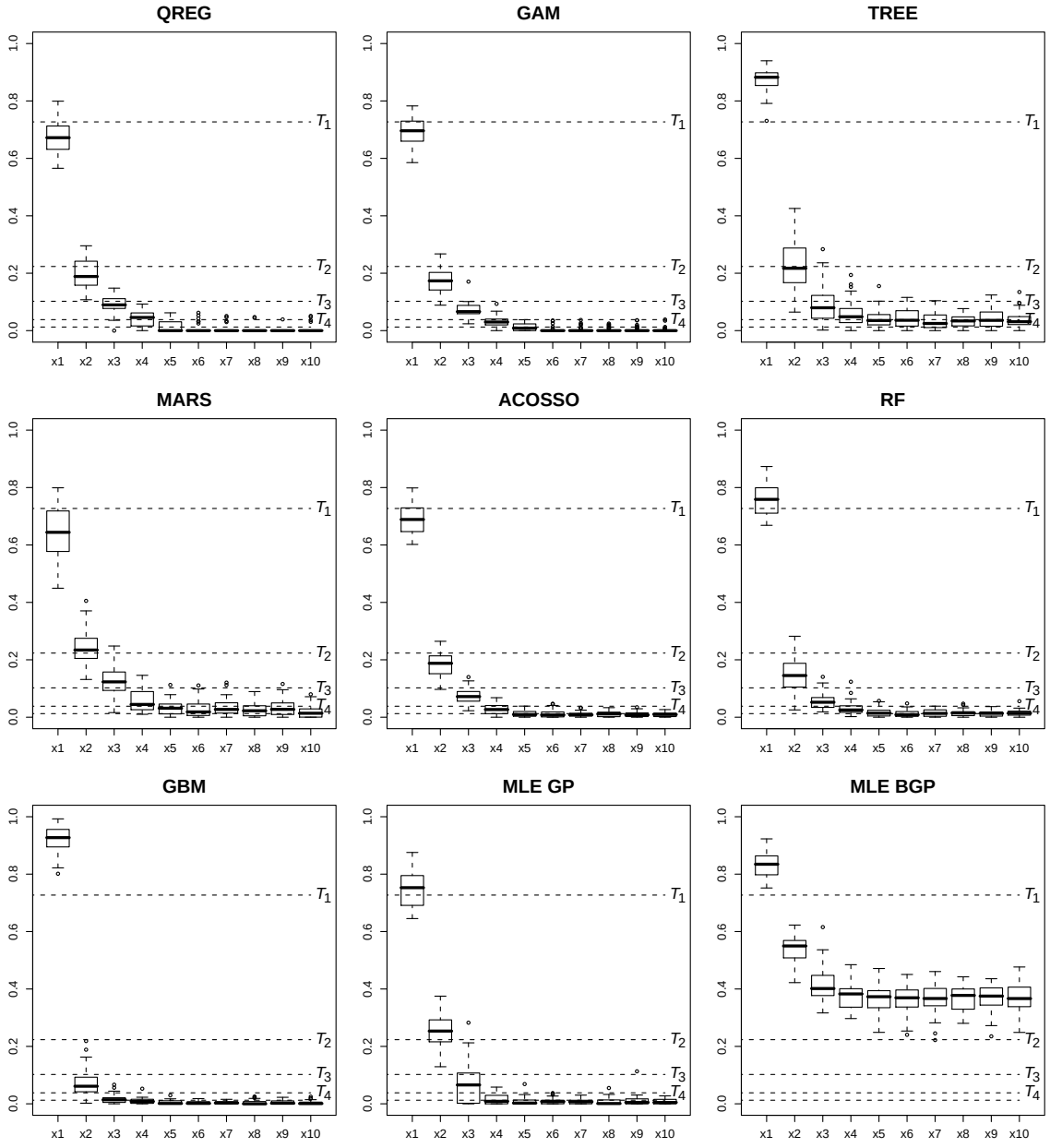


Figure 4: Boxplots of the $\hat{T}_j$ for each of the meta-models for the output y_2. Dashed lines are drawn at the corresponding true values of T_j for reference.

Table 2. Results for y_2 with 100 samples of size $n = 300$.[a]

Var	T_j	RMSE	Cover	Length	Var	T_j	RMSE	Cover	Length
		REG					**QREG**		
x1	0.73	0.66 (0.02)	0.36 (0.08)	0.58 (0.06)	x1	0.73	0.08 (0.01)	0.73 (0.08)	0.19 (0.00)
x2	0.22	0.29 (0.04)	0.97 (0.03)	0.58 (0.05)	x2	0.22	0.05 (0.00)	0.76 (0.08)	0.15 (0.00)
x3	0.10	0.24 (0.04)	1.00 (0.00)	0.67 (0.05)	x3	0.10	0.03 (0.00)	0.88 (0.06)	0.12 (0.00)
x4	0.04	0.22 (0.07)	0.97 (0.03)	0.57 (0.05)	x4	0.04	0.03 (0.00)	0.85 (0.06)	0.09 (0.01)
x5	0.01	0.29 (0.07)	0.94 (0.04)	0.58 (0.05)	x5	0.01	0.02 (0.00)	0.94 (0.04)	0.05 (0.01)
x6	0.00	0.20 (0.07)	1.00 (0.00)	0.51 (0.04)	x6	0.00	0.02 (0.00)	0.97 (0.03)	0.05 (0.00)
x7	0.00	0.08 (0.04)	0.97 (0.03)	0.46 (0.04)	x7	0.00	0.01 (0.00)	0.91 (0.05)	0.04 (0.00)
x8	0.00	0.34 (0.08)	0.94 (0.04)	0.60 (0.05)	x8	0.00	0.01 (0.00)	0.91 (0.05)	0.03 (0.00)
x9	0.00	0.34 (0.08)	0.97 (0.03)	0.54 (0.05)	x9	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)
x10	0.00	0.16 (0.05)	1.00 (0.00)	0.53 (0.05)	x10	0.00	0.01 (0.00)	0.97 (0.03)	0.05 (0.00)
avg.		0.40 (0.08)	0.78 (0.12)	0.61 (0.02)	avg.		0.06 (0.01)	0.79 (0.03)	0.15 (0.01)
		GAM					**RPART**		
x1	0.73	0.06 (0.01)	0.94 (0.04)	0.19 (0.00)	x1	0.73	0.15 (0.01)	0.06 (0.04)	0.12 (0.00)
x2	0.22	0.06 (0.01)	0.73 (0.08)	0.15 (0.00)	x2	0.22	0.09 (0.01)	0.64 (0.08)	0.21 (0.01)
x3	0.10	0.04 (0.00)	0.64 (0.08)	0.10 (0.00)	x3	0.10	0.06 (0.01)	0.79 (0.07)	0.15 (0.01)
x4	0.04	0.02 (0.00)	1.00 (0.00)	0.08 (0.00)	x4	0.04	0.05 (0.01)	0.88 (0.06)	0.12 (0.01)
x5	0.01	0.01 (0.00)	0.97 (0.03)	0.06 (0.00)	x5	0.01	0.04 (0.01)	0.91 (0.05)	0.12 (0.01)
x6	0.00	0.01 (0.00)	0.91 (0.05)	0.04 (0.00)	x6	0.00	0.05 (0.01)	0.79 (0.07)	0.12 (0.01)
x7	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)	x7	0.00	0.04 (0.00)	0.94 (0.04)	0.12 (0.00)
x8	0.00	0.01 (0.00)	0.85 (0.06)	0.03 (0.00)	x8	0.00	0.04 (0.00)	1.00 (0.00)	0.11 (0.00)
x9	0.00	0.01 (0.00)	0.88 (0.06)	0.03 (0.00)	x9	0.00	0.05 (0.01)	0.88 (0.06)	0.12 (0.01)
x10	0.00	0.01 (0.00)	0.91 (0.05)	0.04 (0.00)	x10	0.00	0.05 (0.01)	0.91 (0.05)	0.11 (0.00)
avg.		0.06 (0.00)	0.77 (0.05)	0.15 (0.01)	avg.		0.10 (0.02)	0.50 (0.13)	0.16 (0.02)
		MARS					**ACOSSO**		
x1	0.73	0.11 (0.01)	0.85 (0.06)	0.26 (0.01)	x1	0.73	0.07 (0.01)	0.79 (0.07)	0.19 (0.01)
x2	0.22	0.05 (0.01)	0.85 (0.06)	0.21 (0.01)	x2	0.22	0.06 (0.01)	0.88 (0.06)	0.17 (0.00)
x3	0.10	0.05 (0.00)	0.88 (0.06)	0.18 (0.01)	x3	0.10	0.04 (0.00)	0.91 (0.05)	0.14 (0.01)
x4	0.04	0.04 (0.00)	1.00 (0.00)	0.16 (0.01)	x4	0.04	0.02 (0.00)	0.79 (0.07)	0.08 (0.01)
x5	0.01	0.04 (0.00)	0.97 (0.03)	0.13 (0.01)	x5	0.01	0.01 (0.00)	1.00 (0.00)	0.05 (0.00)
x6	0.00	0.04 (0.01)	0.97 (0.03)	0.13 (0.01)	x6	0.00	0.02 (0.00)	0.94 (0.04)	0.05 (0.00)
x7	0.00	0.04 (0.01)	0.94 (0.04)	0.12 (0.01)	x7	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)
x8	0.00	0.03 (0.00)	0.97 (0.03)	0.14 (0.02)	x8	0.00	0.01 (0.00)	0.97 (0.03)	0.04 (0.00)
x9	0.00	0.04 (0.00)	0.97 (0.03)	0.15 (0.02)	x9	0.00	0.01 (0.00)	0.94 (0.04)	0.04 (0.00)
x10	0.00	0.03 (0.00)	1.00 (0.00)	0.12 (0.01)	x10	0.00	0.01 (0.00)	1.00 (0.00)	0.04 (0.00)
avg.		0.07 (0.01)	0.86 (0.01)	0.22 (0.01)	avg.		0.05 (0.00)	0.86 (0.02)	0.17 (0.01)
		Random Forest					**GBM**		
x1	0.73	0.06 (0.01)	0.88 (0.06)	0.31 (0.01)	x1	0.73	0.20 (0.01)	0.64 (0.08)	0.31 (0.02)
x2	0.22	0.10 (0.01)	0.76 (0.08)	0.25 (0.01)	x2	0.22	0.16 (0.01)	0.30 (0.08)	0.18 (0.01)
x3	0.10	0.05 (0.00)	0.85 (0.06)	0.17 (0.01)	x3	0.10	0.09 (0.00)	0.30 (0.08)	0.09 (0.01)
x4	0.04	0.02 (0.00)	1.00 (0.00)	0.14 (0.01)	x4	0.04	0.03 (0.00)	1.00 (0.00)	0.09 (0.00)
x5	0.01	0.02 (0.00)	1.00 (0.00)	0.12 (0.01)	x5	0.01	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)
x6	0.00	0.02 (0.00)	1.00 (0.00)	0.11 (0.00)	x6	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.01)
x7	0.00	0.02 (0.00)	1.00 (0.00)	0.11 (0.00)	x7	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)
x8	0.00	0.02 (0.00)	1.00 (0.00)	0.11 (0.01)	x8	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)
x9	0.00	0.02 (0.00)	1.00 (0.00)	0.12 (0.01)	x9	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)
x10	0.00	0.02 (0.00)	1.00 (0.00)	0.11 (0.00)	x10	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)

Table 2. Results for y_2 with 100 samples of size $n = 300$.[a]

Var	T_j	RMSE	Cover	Length	Var	T_j	RMSE	Cover	Length
avg.		0.07 (0.01)	0.83 (0.02)	0.24 (0.02)	avg.		0.15 (0.02)	0.41 (0.06)	0.19 (0.04)
		MLE GP					**MLE BGP**		
x1	0.73	0.07 (0.01)	0.79 (0.07)	0.18 (0.01)	x1	0.73	0.12 (0.01)	0.21 (0.07)	0.15 (0.00)
x2	0.22	0.06 (0.01)	0.88 (0.06)	0.21 (0.01)	x2	0.22	0.32 (0.01)	0.00 (0.00)	0.19 (0.01)
x3	0.10	0.07 (0.00)	0.58 (0.09)	0.12 (0.01)	x3	0.10	0.32 (0.01)	0.00 (0.00)	0.21 (0.01)
x4	0.04	0.03 (0.00)	0.94 (0.04)	0.08 (0.01)	x4	0.04	0.34 (0.01)	0.00 (0.00)	0.20 (0.01)
x5	0.01	0.01 (0.00)	1.00 (0.00)	0.06 (0.00)	x5	0.01	0.36 (0.01)	0.00 (0.00)	0.22 (0.00)
x6	0.00	0.01 (0.00)	0.91 (0.05)	0.05 (0.00)	x6	0.00	0.37 (0.01)	0.00 (0.00)	0.21 (0.01)
x7	0.00	0.01 (0.00)	0.94 (0.04)	0.05 (0.00)	x7	0.00	0.38 (0.01)	0.00 (0.00)	0.21 (0.01)
x8	0.00	0.01 (0.00)	0.94 (0.04)	0.06 (0.00)	x8	0.00	0.37 (0.01)	0.00 (0.00)	0.22 (0.01)
x9	0.00	0.01 (0.00)	0.97 (0.03)	0.06 (0.00)	x9	0.00	0.37 (0.01)	0.00 (0.00)	0.22 (0.01)
x10	0.00	0.01 (0.00)	0.94 (0.04)	0.05 (0.00)	x10	0.00	0.37 (0.01)	0.00 (0.00)	0.23 (0.01)
avg.		0.07 (0.00)	0.75 (0.05)	0.17 (0.02)	avg.		0.25 (0.04)	0.07 (0.04)	0.18 (0.01)

[a] Table structure same as in Table 1.

GAM is also one of the best methods on this example. It has the second smallest RMSE averaged across the important inputs (0.06 tied with QREG). It also has good coverages for all of the inputs with the exception of T_2 and T_3. It appears that some of the interactions involving x_2 and x_3 may be biasing these CIs.

RPART is not nearly as effective in this example as in the previous example. The average RMSE (over the important inputs) for RPART is almost 2 times that of QREG, GAM and ACOSSO. RPART also has very poor coverage for T_1 (0.06).

MARS does fairly well again relative to the other methods in this case. It has a little higher RMSE when estimating T_1 and the uninformative T_j's than some of the other methods. However, the coverages for the CIs from MARS are again the most respectable overall as they are all above 0.85.

ACOSSO also has very good overall performance again. It has the lowest average RMSE for the important variables once again. The coverages are also good in general although they are somewhat low for T_1 (0.79) and T_4 (0.79). The interval lengths are also smaller than those for MARS and similar to the lengths for QREG and GAM.

RF has much better performance in this example compared to the previous example. Random Forest gives results similar to MARS for RMSE and coverage, but also has notably larger CI lengths. GBM gives disappointing results for RMSE (average on important variables of 3 times that for ACOSSO) and coverage (as low as 0.30 for T_2 and T_3), even though the CI lengths are large relative to other methods.

MLE GP again has good performance for estimating T_j. It has RMSE and CI lengths comparable to the other top methods on this example (QREG, GAM, MARS, ACOSSO). MLE GP also has reasonable coverage for all T_j but T_3 (0.58). MLE BGP has very poor performance on this example with RMSE for uninformative variables 30 times higher than those for MLE GP. This is a case where it is evident that the mean of a function of a random variable is not equal to that function evaluated at the

mean. This is essentially the difference between the T_j estimates for MLE GP ($\hat{T}_j$) and MLE BGP ($\tilde{T}_j$). MLE BGP evaluates T_j at several posterior realizations of f, then averages these T_j to obtain $\tilde{T}_j$. MLE GP evaluates T_j at $\hat{f}$ (the posterior average value of f). In this example, the posterior mean for f shows very little change across the uninformative variables giving $\hat{T}_j$ near 0 for most of the simulations. However, the posterior distribution of f has a fair amount of fluctuation around 0 in the amount of signal across uninformative variables, leading to a $\tilde{T}_j$ substantially different from 0. This example is a fairly pathological case for the Bayes framework since the function f_2 is a "V" as seen in Figure 3. The prior distribution imposed on f by the GP (with powered exponential covariance) is stationary, meaning the unconditional variance is the same in all regions of the domain. Thus the GP prior assumes a similar amount of continuity and change over the entire domain, making it a poor choice for this situation.

4.3 Output y_3

The last analytic output is the non-monotonic Ishigami function; see Figures 5 and 6. Here, the difficulty in estimation lies in the periodic nature of the relationship between y_3 and x_2 and in the lack of any main effect due to x_3.

The results y_3 are presented in Table 3. This proved to be the most complex and challenging test problem for many of the methods. REG has high RMSE as would be expected, but QREG has average RMSE nearly as high in this case (0.28) and very low coverages (0.06 for T_2). This is due to the inability of QREG to model the periodic behavior that is evident across x_2. GAM is better than QREG in terms of average RMSE but its CI coverages are also very poor in general (0% for T_3).

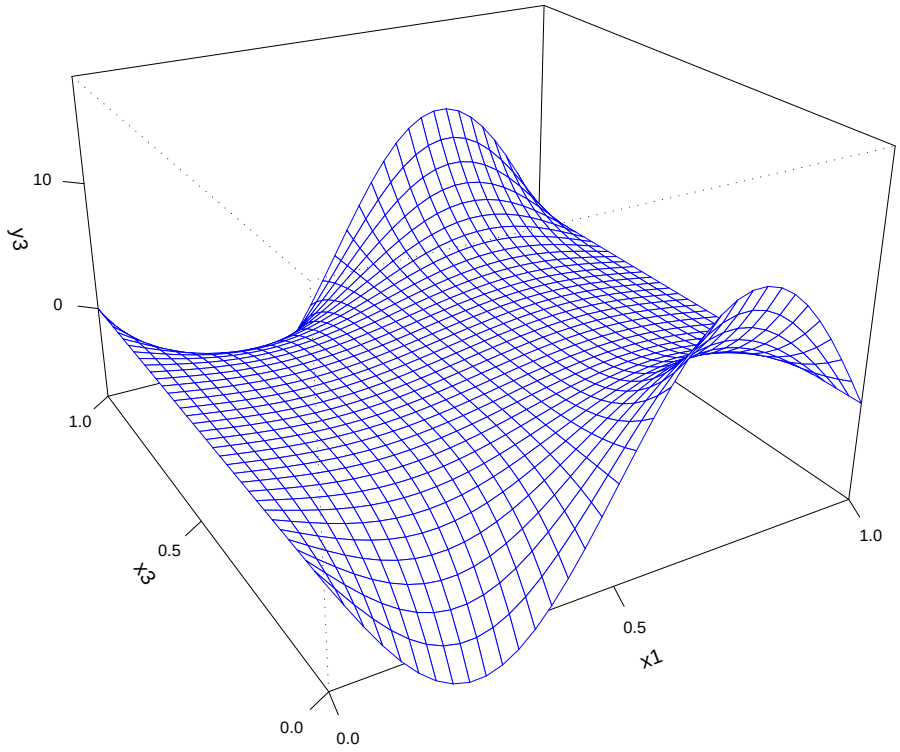
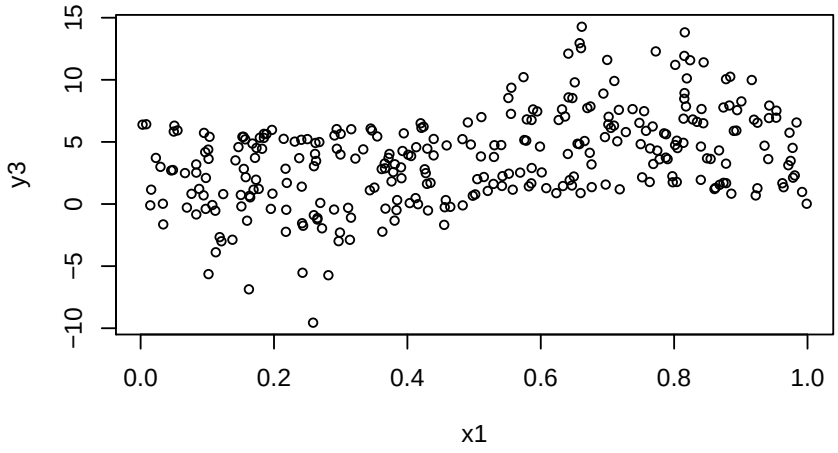
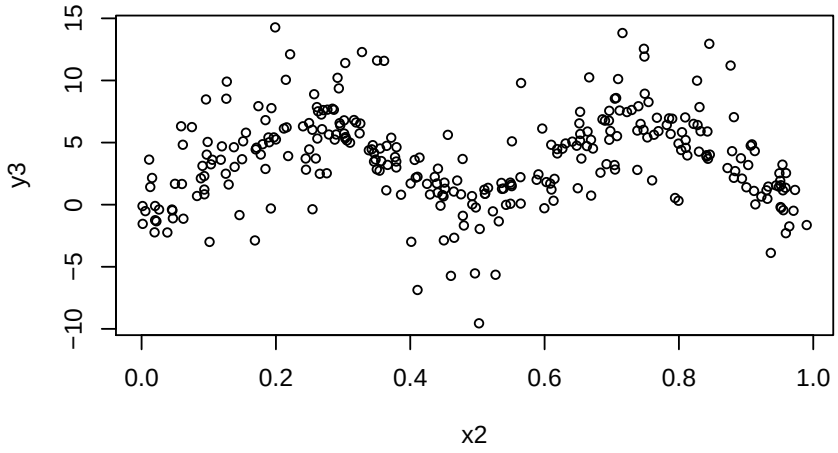


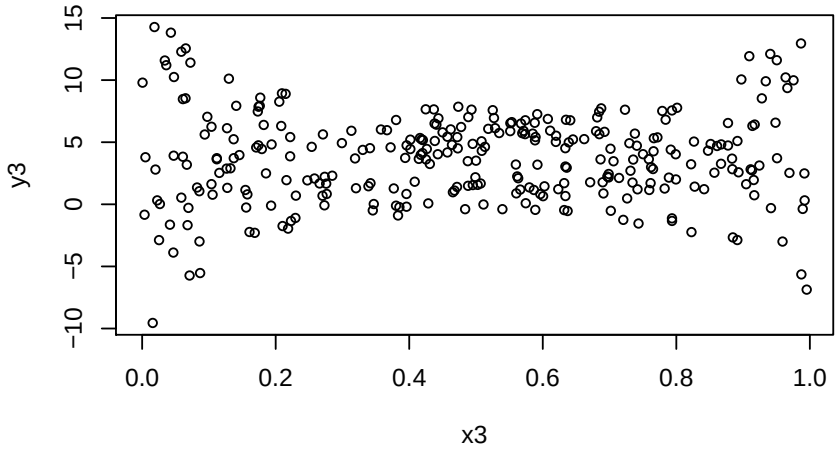
Figure 5: Plot of output y_3 against inputs x_1 and x_3 with x_2 fixed at 0.5.



(a)



(b)



(c)

Figure 6: Scatterplots of output y_3 against inputs x_1, x_2, and x_3 individually.

Table 3. Results for y_3 with 100 samples of size $n = 300$.[a]

Var	T_j	RMSE	Cover	Length	Var	T_j	RMSE	Cover	Length
		REG					**QREG**		
x1	0.55	0.36 (0.01)	0.11 (0.05)	0.30 (0.02)	x1	0.55	0.27 (0.02)	0.31 (0.08)	0.37 (0.01)
x2	0.45	0.44 (0.00)	0.00 (0.00)	0.09 (0.01)	x2	0.45	0.35 (0.01)	0.06 (0.04)	0.26 (0.02)
x3	0.24	0.22 (0.00)	0.06 (0.04)	0.11 (0.01)	x3	0.24	0.20 (0.01)	0.34 (0.08)	0.20 (0.02)
x4	0.00	0.04 (0.02)	0.97 (0.03)	0.09 (0.01)	x4	0.00	0.06 (0.01)	1.00 (0.00)	0.16 (0.01)
x5	0.00	0.02 (0.00)	1.00 (0.00)	0.08 (0.01)	x5	0.00	0.05 (0.01)	0.97 (0.03)	0.16 (0.01)
x6	0.00	0.03 (0.01)	1.00 (0.00)	0.08 (0.01)	x6	0.00	0.06 (0.02)	0.94 (0.04)	0.15 (0.01)
x7	0.00	0.04 (0.01)	1.00 (0.00)	0.10 (0.01)	x7	0.00	0.05 (0.01)	0.97 (0.03)	0.16 (0.01)
x8	0.00	0.02 (0.00)	1.00 (0.00)	0.08 (0.01)	x8	0.00	0.02 (0.01)	1.00 (0.00)	0.14 (0.01)
x9	0.00	0.04 (0.01)	0.97 (0.03)	0.09 (0.01)	x9	0.00	0.06 (0.02)	0.97 (0.03)	0.15 (0.01)
x10	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.01)	x10	0.00	0.03 (0.01)	1.00 (0.00)	0.14 (0.01)
avg.		0.34 (0.04)	0.06 (0.02)	0.17 (0.04)	avg.		0.28 (0.03)	0.24 (0.05)	0.28 (0.03)
		GAM					**RPART**		
x1	0.55	0.15 (0.01)	0.09 (0.05)	0.17 (0.01)	x1	0.55	0.07 (0.01)	0.80 (0.07)	0.20 (0.00)
x2	0.45	0.12 (0.01)	0.14 (0.06)	0.17 (0.00)	x2	0.45	0.08 (0.01)	0.80 (0.07)	0.25 (0.01)
x3	0.24	0.23 (0.00)	0.00 (0.00)	0.05 (0.00)	x3	0.24	0.13 (0.01)	0.29 (0.08)	0.20 (0.01)
x4	0.00	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)	x4	0.00	0.01 (0.00)	1.00 (0.00)	0.06 (0.00)
x5	0.00	0.00 (0.00)	0.91 (0.05)	0.03 (0.00)	x5	0.00	0.02 (0.00)	0.94 (0.04)	0.07 (0.00)
x6	0.00	0.00 (0.00)	0.94 (0.04)	0.02 (0.00)	x6	0.00	0.02 (0.00)	0.94 (0.04)	0.06 (0.00)
x7	0.00	0.01 (0.00)	0.91 (0.05)	0.03 (0.00)	x7	0.00	0.03 (0.01)	0.86 (0.06)	0.07 (0.00)
x8	0.00	0.01 (0.00)	0.89 (0.05)	0.02 (0.00)	x8	0.00	0.02 (0.00)	0.97 (0.03)	0.07 (0.00)
x9	0.00	0.01 (0.00)	0.86 (0.06)	0.02 (0.00)	x9	0.00	0.01 (0.00)	1.00 (0.00)	0.06 (0.00)
x10	0.00	0.00 (0.00)	0.91 (0.05)	0.03 (0.00)	x10	0.00	0.02 (0.00)	0.94 (0.04)	0.06 (0.00)
avg.		0.17 (0.02)	0.08 (0.02)	0.13 (0.02)	avg.		0.09 (0.01)	0.63 (0.10)	0.22 (0.01)
		MARS					**ACOSSO**		
x1	0.55	0.10 (0.04)	0.89 (0.05)	0.19 (0.01)	x1	0.55	0.06 (0.00)	0.74 (0.07)	0.14 (0.00)
x2	0.45	0.09 (0.04)	0.94 (0.04)	0.20 (0.02)	x2	0.45	0.05 (0.00)	0.71 (0.08)	0.14 (0.00)
x3	0.24	0.11 (0.05)	0.94 (0.04)	0.17 (0.02)	x3	0.24	0.07 (0.00)	0.83 (0.06)	0.11 (0.00)
x4	0.00	0.01 (0.00)	0.97 (0.03)	0.08 (0.02)	x4	0.00	0.01 (0.00)	0.97 (0.03)	0.02 (0.00)
x5	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.02)	x5	0.00	0.00 (0.00)	0.97 (0.03)	0.02 (0.00)
x6	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.02)	x6	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
x7	0.00	0.02 (0.01)	0.94 (0.04)	0.08 (0.02)	x7	0.00	0.01 (0.00)	1.00 (0.00)	0.02 (0.00)
x8	0.00	0.04 (0.02)	0.97 (0.03)	0.08 (0.02)	x8	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
x9	0.00	0.02 (0.00)	0.91 (0.05)	0.07 (0.01)	x9	0.00	0.01 (0.00)	0.97 (0.03)	0.02 (0.00)
x10	0.00	0.02 (0.00)	1.00 (0.00)	0.08 (0.02)	x10	0.00	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)
avg.		0.10 (0.00)	0.92 (0.01)	0.19 (0.01)	avg.		0.06 (0.00)	0.76 (0.02)	0.13 (0.00)
		Random Forest					**GBM**		
x1	0.55	0.18 (0.01)	0.49 (0.08)	0.21 (0.01)	x1	0.55	0.29 (0.01)	0.17 (0.06)	0.28 (0.02)
x2	0.45	0.22 (0.01)	0.43 (0.08)	0.18 (0.01)	x2	0.45	0.31 (0.01)	0.17 (0.06)	0.16 (0.02)
x3	0.24	0.14 (0.00)	0.26 (0.07)	0.11 (0.00)	x3	0.24	0.09 (0.01)	0.60 (0.08)	0.17 (0.01)
x4	0.00	0.02 (0.00)	0.94 (0.04)	0.08 (0.00)	x4	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)
x5	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)	x5	0.00	0.01 (0.00)	0.97 (0.03)	0.08 (0.00)
x6	0.00	0.02 (0.00)	1.00 (0.00)	0.08 (0.00)	x6	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)
x7	0.00	0.02 (0.00)	0.97 (0.03)	0.08 (0.00)	x7	0.00	0.01 (0.00)	0.97 (0.03)	0.07 (0.00)
x8	0.00	0.01 (0.00)	0.94 (0.04)	0.07 (0.00)	x8	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)
x9	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)	x9	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.01)
x10	0.00	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)	x10	0.00	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)

Table 3. Results for y_3 with 100 samples of size $n = 300$.[a]

Var	T_j	RMSE	Cover	Length	Var	T_j	RMSE	Cover	Length
avg.		0.18 (0.01)	0.39 (0.04)	0.17 (0.02)	avg.		0.23 (0.04)	0.31 (0.08)	0.20 (0.02)
		MLE GP					**MLE BGP**		
x1	0.55	0.08 (0.02)	0.83 (0.06)	0.28 (0.01)	x1	0.55	0.05 (0.01)	0.91 (0.05)	0.15 (0.00)
x2	0.45	0.08 (0.01)	0.83 (0.06)	0.25 (0.01)	x2	0.45	0.12 (0.02)	0.60 (0.08)	0.16 (0.00)
x3	0.24	0.09 (0.02)	0.69 (0.08)	0.21 (0.01)	x3	0.24	0.04 (0.01)	0.97 (0.03)	0.16 (0.00)
x4	0.00	0.01 (0.00)	0.89 (0.05)	0.05 (0.00)	x4	0.00	0.10 (0.02)	0.86 (0.06)	0.12 (0.01)
x5	0.00	0.01 (0.00)	0.94 (0.04)	0.05 (0.00)	x5	0.00	0.10 (0.02)	0.86 (0.06)	0.12 (0.01)
x6	0.00	0.01 (0.00)	0.89 (0.05)	0.05 (0.00)	x6	0.00	0.10 (0.02)	0.83 (0.06)	0.12 (0.01)
x7	0.00	0.01 (0.00)	0.94 (0.04)	0.05 (0.00)	x7	0.00	0.10 (0.02)	0.83 (0.06)	0.13 (0.01)
x8	0.00	0.01 (0.00)	1.00 (0.00)	0.06 (0.00)	x8	0.00	0.10 (0.02)	0.86 (0.06)	0.12 (0.01)
x9	0.00	0.02 (0.01)	0.89 (0.05)	0.06 (0.00)	x9	0.00	0.11 (0.02)	0.83 (0.06)	0.13 (0.01)
x10	0.00	0.01 (0.00)	0.94 (0.04)	0.05 (0.00)	x10	0.00	0.10 (0.02)	0.86 (0.06)	0.13 (0.01)
avg.		0.08 (0.00)	0.78 (0.03)	0.25 (0.01)	avg.		0.07 (0.02)	0.83 (0.07)	0.16 (0.00)

[a] Table structure same as in Table 1.

RPART is one of the better methods for RMSE and coverages are much better than GAM or QREG in general, although there is a very low coverage for T_3 (0.29).

MARS has reasonable RMSE for the T_j; slightly higher than those for ACOSSO and MLE GP, but better than most other methods. Once again, the coverage of the CIs produced from MARS are very close to the 95% level though for all of the inputs. MARS is by far the most consistent method for maintaining the nominal coverages. The CI lengths for MARS are larger than ACOSSO but smaller than those of MLE GP.

ACOSSO is substantially better than all other methods in terms of RMSE. The coverage for ACOSSO is below nominal for the important variables (0.74 for T_1 0.71 for T_2, and 0.83 for T_3), but is once again at or above 0.95 for the uninformative variables. The CI lengths are also the shortest of any method.

Random Forest has high RMSE for the three important variables, but has very low RMSE for the uninformative inputs. The coverage is also very low for the three important variables. GBM is again similar to Random Forest in that the estimates and coverages are very good for the uninformative variables but poor for the important ones.

MLE GP is one of the better methods along with MARS and ACOSSO in this example. RMSE for the T_j is lower than MARS but higher than ACOSSO in general. The coverages are reasonable with the exception of that for T_3 (0.69). Unfortunately, the CI lengths are a bit long though (0.28 for T_1 and 0.25 for T_2).

MLE BGP has more success here than in the previous example. It is one of the better methods insofar as RMSE of T_j on the important variables. However, it once again struggles to estimate the T_j for the uninformative variables (RMSE ≥ 0.10 for all uninformative inputs). In addition, the coverages are not nearly as good as with other methods (0.85 for all of the uninformative inputs).

Figure 4 displays the boxplots for each meta model of the $\hat{T}_j$ distribution, $j = 1, \ldots, 10$ for the output y_3. These plots indicate the difficulty that many of the meta-models have with this output. The QREG plot makes it clear that QREG is not capable of modeling this output and similarly for GAM. TREE has reasonable performance for $\hat{T}_1$, and $\hat{T}_2$ and the $\hat{T}_j$ for the unimportant inputs ($j > 3$), but $\hat{T}_3$ is biased high. MARS has very good performance overall on this output. MARS has very tightly centered distributions for $\hat{T}_j$ around the true T_j, but there all also several outlying values (e.g. notice the two very low points $\hat{T}_1 = 0.0$ and $\hat{T}_1 = 0.05$) indicating that MARS is not always the most stable. ACOSSO also has good performance, but it too has a couple outlying $\hat{T}_j$ (one very high $\hat{T}_1$ and one very low $\hat{T}_2$). RF and GBM both have a sizable bias for $\hat{T}_1$ - $\hat{T}_3$. MLE GP has sound performance, very similar to ACOSSO, but the outlying $\hat{T}_j$ are not as extreme. MLE BGP is similar to MLE GP for $\hat{T}_1$ - $\hat{T}_3$ but has upward bias with many outlying $\hat{T}_j$ for the unimportant variables ($j > 3$).

4.4 Effect of Sample Size

Here we summarize the various methods when applied to the preceding examples with samples sizes of $n = 75$, $n = 150$, and $n = 300$. The CIs based on the bootstrap have asymptotically correct coverages under certain regularity conditions (p. 37-39 [20]). An exact treatment of these conditions for the T_j CIs is beyond the scope of this presentation. It is, however, necessary that the meta-model used be a consistent estimator of the true function (i.e. $\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})$ converges in probability to 0 for each $\boldsymbol{x}$). Hence, we cannot expect the bootstrap CIs to have the correct coverage unless the meta-model is appropriate and until we have a sufficiently large sample. The purpose of the following exercise is to answer the following two questions: (i) What sample size is necessary for bootstrap CIs to be useful?, and (ii) Are the coverages increasing to the nominal (i.e. 95%) level as sample size increases? The values presented in the following tables for each method are the averages across input variables of the respective summary statistics.

Table 4 contains the results for output y_1 averaged over only the *important* inputs for the various sample sizes. We define an input to be important for an output if $T_j \geq 0.1$. Notice that the coverages for REG and QREG actually go down slightly as sample size increases. This is to be expected since neither of these meta-models is a consistent estimator for the underlying true function f_1 in this case. For instance, the function is not quadratic but the estimate $\hat{f}$ from QREG is and the estimate will have less and less variability the larger the sample size. Hence, with REG and QREG, the larger sample size results in shorter length intervals around a biased quantity. In fact, we might expect these coverages to decrease to zero as sample size continues to be increased. However, for practical purposes QREG is producing CIs that are useful for all sample sizes considered.

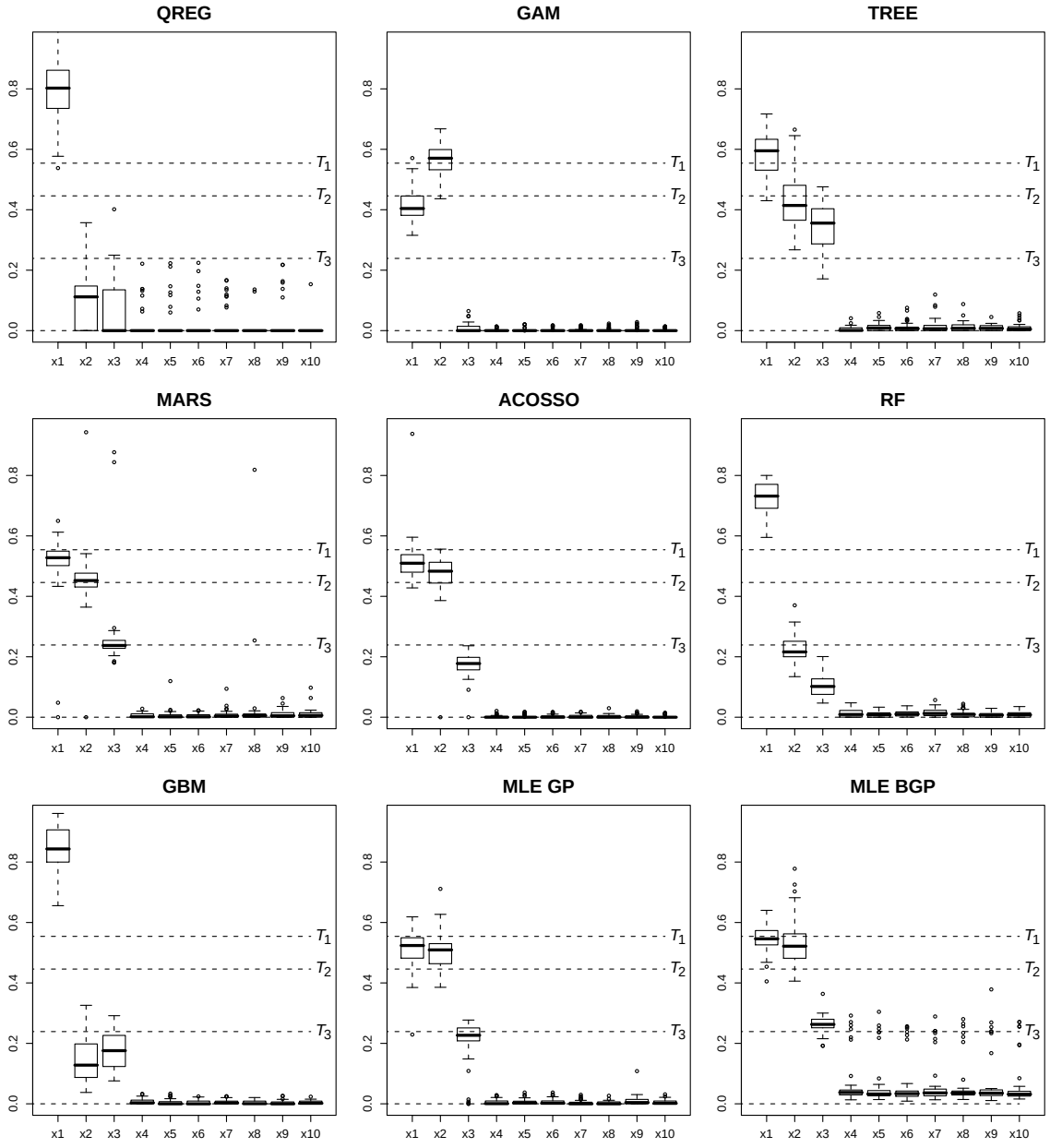


Figure 7: Boxplots of the $\hat{T}_j$ for each of the meta-models for the output y_3. Dashed lines are drawn at the corresponding true values of T_j for reference.

Table 4: Results for y_1 averaged across the *important* inputs with various sample sizes.

n^a	RMSE[b]	Cover[c]	Length[d]	n	RMSE	Cover	Length
REG				**QREG**			
75	0.15 (0.03)	0.70 (0.19)	0.28 (0.00)	75	0.07 (0.01)	0.79 (0.05)	0.18 (0.01)
150	0.13 (0.02)	0.47 (0.22)	0.22 (0.00)	150	0.04 (0.00)	0.92 (0.02)	0.14 (0.00)
300	0.12 (0.03)	0.40 (0.28)	0.15 (0.00)	300	0.04 (0.00)	0.79 (0.09)	0.10 (0.00)
GAM				**RPART**			
75	0.16 (0.02)	0.21 (0.05)	0.26 (0.01)	75	0.17 (0.01)	0.62 (0.06)	0.33 (0.03)
150	0.14 (0.01)	0.10 (0.00)	0.20 (0.00)	150	0.05 (0.00)	0.95 (0.00)	0.19 (0.02)
300	0.13 (0.02)	0.03 (0.02)	0.15 (0.00)	300	0.04 (0.01)	0.80 (0.08)	0.11 (0.01)
MARS				**ACOSSO**			
75	0.07 (0.01)	0.89 (0.03)	0.32 (0.00)	75	0.05 (0.01)	0.89 (0.00)	0.19 (0.01)
150	0.16 (0.04)	0.97 (0.02)	0.26 (0.00)	150	0.03 (0.00)	0.95 (0.00)	0.13 (0.00)
300	0.02 (0.00)	0.96 (0.01)	0.11 (0.00)	300	0.02 (0.00)	0.89 (0.02)	0.10 (0.00)
Random Forest				**GBM**			
75	0.18 (0.06)	0.73 (0.19)	0.43 (0.04)	75	0.22 (0.06)	0.66 (0.16)	0.40 (0.03)
150	0.13 (0.04)	0.79 (0.15)	0.29 (0.01)	150	0.11 (0.02)	0.95 (0.00)	0.35 (0.01)
300	0.10 (0.04)	0.83 (0.08)	0.17 (0.00)	300	0.07 (0.00)	0.91 (0.06)	0.23 (0.02)
MLE GP				**MLE BGP**			
75	0.05 (0.00)	0.79 (0.00)	0.16 (0.01)	75	0.05 (0.00)	0.93 (0.03)	0.20 (0.00)
150	0.03 (0.00)	0.87 (0.02)	0.14 (0.00)	150	0.03 (0.00)	1.00 (0.00)	0.20 (0.00)
300	0.03 (0.00)	0.83 (0.04)	0.12 (0.00)	300	0.03 (0.00)	1.00 (0.00)	0.20 (0.00)

[a] Sample size used to generate the data.

[b] Average across inputs with $T_j \geq 0.10$ of the RMSE of $\hat{T}_j$ given by the corresponding meta-model; estimated standard deviation of this average RRMSE given in parantheses.

[c] Average across inputs with $T_j \geq 0.10$ of the coverage of CI's generated by the corresponding meta-model; estimated standard deviation of this average coverage given in parantheses.

[d] Average across inputs with $T_j \geq 0.10$ of the length of CI's generated by the corresponding meta-model; estimated standard deviation of this average length given in parantheses.

GAM has a similar issue here as well. The true function is not additive, so a GAM model will always have a bias that does not go away as sample size increases. MARS and ACOSSO both have good coverage at all sample sizes. Notice that the average interval length is decreasing for these methods as sample size increases. However, the RMSE is also decreasing fast enough to allow for the shorter intervals to maintain close to the 95% coverage. Random Forest does not have coverages as near the nominal level but they are increasing with sample size. The CI length is also decreasing for Random Forest. For GBM the coverage increases to near the nominal level and the CI length decreases as sample size increases, but not as much as with other methods. MLE GP appears to have an increasing trend in coverage and decreasing interval width with increasing sample size. However, the coverage does go down slightly when moving from $n = 150$ to $n = 300$. For MLE BGP, the coverage is quite good at all sample sizes but the CI width does not appear to be decreasing like the other methods.

Table 5 contains the results for output y_1 averaged over only the *unimportant* inputs for the various sample sizes. We define an input to be unimportant for an output if $T_j < 0.01$. Notice that important inputs are defined to have $T_j > 0.10$, which allows some inputs to be in a grey area where they are neither unimportant nor important. All of the methods seem to have good coverages at all sample sizes for the unimportant inputs. The CI lengths are deceasing for most methods, but this decrease is much less dramatic than with CI length for the important inputs since the CI lengths for the unimportant inputs are generally much smaller.

Summaries of results for y_2 and the three sample sizes for the important inputs are presented in Table 6. The CIs based on REG are not useful here. The average length is greater then 0.5 for each sample size making these CIs too wide to be informative. Both GAM and QREG seem to struggle slightly with the interactions involved in this example as the coverages for both trend downward with increasing sample size. However, with coverages around 0.80, the CIs from QREG and GAM are still usable. The RMSE and CI lengths are decreasing with increase in sample size. GAM and QREG have among the shortest length CIs for any of the methods.

The coverages for RPART decline substantially as sample size is increased from $n = 75$ to $n = 300$ (0.70 to 0.50). This is a particularly difficult function for RPART to model since the "V" shape is not well approximated by piecewise constants. Still, the decreasing coverage is somewhat unsettling. MARS does not have increasing coverage but it remains relatively high (≥ 0.85) for all sample sizes and also has decreasing RMSE and CI length as sample size increases. ACOSSO has increasing coverage as sample size increases $n = 75$ to $n = 300$ (0.81 to 0.86). In addition the RMSE and CI lengths are decreasing to both become very small for $n = 300$.

Random Forest has good coverages for all sample sizes but declines a bit as sample size increases $n = 75$ to $n = 300$ (0.92 to 0.83). RMSE and CI lengths are decreasing so that Random Forest appears to be performing reasonably well. GBM has a substantial drop in coverage with increasing sample size (0.94 to 0.41). This gives some cause for concern when using GBM for CIs.

Table 5: Results for y_1 averaged across the *unimportant* inputs with various sample sizes.

n^a	RMSE[b]	Cover[c]	Length[d]	n	RMSE	Cover	Length
REG				**QREG**			
75	0.01 (0.00)	1.00 (0.00)	0.04 (0.00)	75	0.01 (0.00)	0.98 (0.00)	0.03 (0.00)
150	0.01 (0.00)	1.00 (0.00)	0.03 (0.00)	150	0.00 (0.00)	0.99 (0.00)	0.02 (0.00)
300	0.00 (0.00)	1.00 (0.00)	0.02 (0.00)	300	0.00 (0.00)	0.99 (0.00)	0.02 (0.00)
GAM				**RPART**			
75	0.02 (0.00)	0.90 (0.01)	0.06 (0.00)	75	0.01 (0.00)	1.00 (0.00)	0.07 (0.00)
150	0.01 (0.00)	0.95 (0.01)	0.04 (0.00)	150	0.00 (0.00)	0.99 (0.00)	0.05 (0.00)
300	0.01 (0.00)	0.95 (0.01)	0.03 (0.00)	300	0.00 (0.00)	1.00 (0.00)	0.04 (0.00)
MARS				**ACOSSO**			
75	0.02 (0.00)	1.00 (0.00)	0.13 (0.00)	75	0.01 (0.00)	0.98 (0.00)	0.04 (0.00)
150	0.06 (0.01)	0.98 (0.00)	0.16 (0.00)	150	0.01 (0.00)	0.99 (0.00)	0.04 (0.00)
300	0.01 (0.00)	0.96 (0.00)	0.08 (0.00)	300	0.01 (0.00)	0.97 (0.00)	0.04 (0.00)
Random Forest				**GBM**			
75	0.02 (0.00)	1.00 (0.00)	0.14 (0.00)	75	0.01 (0.00)	0.98 (0.00)	0.09 (0.00)
150	0.02 (0.00)	0.93 (0.01)	0.07 (0.00)	150	0.01 (0.00)	0.97 (0.00)	0.07 (0.00)
300	0.02 (0.00)	0.90 (0.00)	0.06 (0.00)	300	0.02 (0.00)	0.96 (0.00)	0.08 (0.00)
MLE GP				**MLE BGP**			
75	0.02 (0.00)	0.91 (0.01)	0.09 (0.00)	75	0.04 (0.00)	1.00 (0.00)	0.17 (0.00)
150	0.02 (0.00)	0.91 (0.01)	0.09 (0.00)	150	0.05 (0.00)	1.00 (0.00)	0.17 (0.00)
300	0.02 (0.00)	0.91 (0.01)	0.09 (0.00)	300	0.05 (0.00)	1.00 (0.00)	0.18 (0.00)

[a] Sample size used to generate the data.

[b] Average across inputs with $T_j \leq 0.01$ of the RMSE of $\hat{T}_j$ given by the corresponding meta-model; estimated standard deviation of this average RRMSE given in parantheses.

[c] Average across inputs with $T_j \leq 0.01$ of the coverage of CI's generated by the corresponding meta-model; estimated standard deviation of this average coverage given in parantheses.

[d] Average across inputs with $T_j \leq 0.01$ of the length of CI's generated by the corresponding meta-model; estimated standard deviation of this average length given in parantheses.

Table 6: Results for y_2 averaged across the *important* inputs with various sample sizes.

n	RMSE	Cover	Length	n	RMSE	Cover	Length
REG				**QREG**			
75	0.40 (0.07)	0.74 (0.12)	0.54 (0.01)	75	0.12 (0.01)	0.84 (0.02)	0.31 (0.03)
150	0.40 (0.08)	0.73 (0.14)	0.56 (0.01)	150	0.08 (0.01)	0.80 (0.02)	0.22 (0.01)
300	0.40 (0.08)	0.78 (0.12)	0.61 (0.02)	300	0.06 (0.01)	0.79 (0.03)	0.15 (0.01)
GAM				**RPART**			
75	0.12 (0.02)	0.82 (0.01)	0.29 (0.03)	75	0.18 (0.01)	0.70 (0.05)	0.35 (0.03)
150	0.08 (0.01)	0.79 (0.03)	0.21 (0.02)	150	0.11 (0.01)	0.59 (0.12)	0.22 (0.01)
300	0.06 (0.00)	0.77 (0.05)	0.15 (0.01)	300	0.10 (0.02)	0.50 (0.13)	0.16 (0.02)
MARS				**ACOSSO**			
75	0.13 (0.01)	0.87 (0.02)	0.39 (0.03)	75	0.28 (0.04)	0.81 (0.04)	0.53 (0.03)
150	0.10 (0.01)	0.85 (0.03)	0.28 (0.02)	150	0.11 (0.00)	0.75 (0.03)	0.23 (0.02)
300	0.07 (0.01)	0.86 (0.01)	0.22 (0.01)	300	0.05 (0.00)	0.86 (0.02)	0.17 (0.01)
Random Forest				**GBM**			
75	0.14 (0.02)	0.92 (0.01)	0.42 (0.04)	75	0.15 (0.01)	0.94 (0.02)	0.47 (0.07)
150	0.09 (0.01)	0.93 (0.04)	0.36 (0.04)	150	0.14 (0.02)	0.82 (0.07)	0.35 (0.07)
300	0.07 (0.01)	0.83 (0.02)	0.24 (0.02)	300	0.15 (0.02)	0.41 (0.06)	0.19 (0.04)
MLE GP				**MLE BGP**			
75	0.14 (0.01)	0.52 (0.07)	0.24 (0.02)	75	0.18 (0.02)	0.53 (0.04)	0.24 (0.01)
150	0.10 (0.01)	0.65 (0.04)	0.21 (0.01)	150	0.24 (0.03)	0.21 (0.05)	0.22 (0.02)
300	0.07 (0.00)	0.75 (0.05)	0.17 (0.02)	300	0.25 (0.04)	0.07 (0.04)	0.18 (0.01)

[a] Table structure same as in Table 4.

MLE GP has low coverage (0.52) at $n = 75$ but increases to (0.75) by $n = 300$. Thus, for larger sample sizes the CIs produced from MLE GP are usable. The RMSE and CI lengths also decrease significantly with increased sample size. MLE BGP really struggles with this example. Coverages decrease from 0.53 when $n = 75$ to 0.07 when $n = 300$. Even more troubling is that RMSE actually increases with sample size. As mentioned, this is a very difficult function to model with a stationary prior distribution, leading to the poor results. However, this is reason to question the use of this procedure in practice when there is a chance of sharp changes in the output.

Table 7 gives the results for output y_2 averaged over only the *unimportant* inputs for the various sample sizes. As in the previous example, the coverages for all the of the methods are quite good, with the exception of MLE BGP, but even this increases with sample size. All of the methods have CI length decreasing with sample size, which is much more pronounced here than for y_1. This is due to the fact that the signal to noise ratio is much lower in this example making it harder to identify the uninformative variables at small sample sizes.

Table 8 contains the results for y_3 and the three sample sizes for the important inputs. The coverages for REG, QREG, and GAM decrease as sample size increases. QREG cannot model the periodic nature of the function, while GAM cannot model the interaction between x_1 and x_3. In either case, the result as sample size increases

Table 7: Results for y_2 averaged across the *unimportant* inputs with various sample sizes.

n	RMSE	Cover	Length	n	RMSE	Cover	Length
REG				**QREG**			
75	0.23 (0.01)	0.98 (0.00)	0.53 (0.00)	75	0.05 (0.00)	0.95 (0.01)	0.15 (0.00)
150	0.26 (0.01)	0.97 (0.00)	0.55 (0.00)	150	0.03 (0.00)	0.96 (0.00)	0.08 (0.00)
300	0.22 (0.02)	0.98 (0.01)	0.53 (0.01)	300	0.01 (0.00)	0.94 (0.01)	0.04 (0.00)
GAM				**RPART**			
75	0.04 (0.00)	0.86 (0.01)	0.08 (0.00)	75	0.07 (0.00)	0.97 (0.00)	0.22 (0.00)
150	0.02 (0.00)	0.90 (0.01)	0.06 (0.00)	150	0.06 (0.00)	0.92 (0.00)	0.16 (0.00)
300	0.01 (0.00)	0.90 (0.01)	0.04 (0.00)	300	0.05 (0.00)	0.90 (0.02)	0.12 (0.00)
MARS				**ACOSSO**			
75	0.07 (0.00)	0.95 (0.01)	0.20 (0.00)	75	0.18 (0.01)	0.94 (0.00)	0.39 (0.01)
150	0.06 (0.00)	0.96 (0.00)	0.16 (0.00)	150	0.03 (0.00)	0.96 (0.01)	0.12 (0.00)
300	0.04 (0.00)	0.97 (0.00)	0.13 (0.00)	300	0.01 (0.00)	0.96 (0.01)	0.04 (0.00)
Random Forest				**GBM**			
75	0.05 (0.00)	1.00 (0.00)	0.29 (0.00)	75	0.03 (0.00)	1.00 (0.00)	0.30 (0.00)
150	0.03 (0.00)	1.00 (0.00)	0.21 (0.00)	150	0.01 (0.00)	1.00 (0.00)	0.16 (0.00)
300	0.02 (0.00)	1.00 (0.00)	0.11 (0.00)	300	0.01 (0.00)	1.00 (0.00)	0.08 (0.00)
MLE GP				**MLE BGP**			
75	0.09 (0.00)	0.81 (0.01)	0.14 (0.00)	75	0.21 (0.00)	0.50 (0.02)	0.25 (0.00)
150	0.03 (0.00)	0.94 (0.01)	0.09 (0.00)	150	0.34 (0.00)	0.02 (0.00)	0.26 (0.00)
300	0.01 (0.00)	0.94 (0.00)	0.06 (0.00)	300	0.37 (0.00)	0.00 (0.00)	0.22 (0.00)

[a] Table structure same as in Table 5.

Table 8: Results for y_3 averaged across the *important* inputs with various sample sizes.

n	RMSE	Cover	Length	n	RMSE	Cover	Length
REG				**QREG**			
75	0.30 (0.04)	0.49 (0.14)	0.40 (0.08)	75	0.30 (0.03)	0.62 (0.12)	0.47 (0.05)
150	0.33 (0.04)	0.24 (0.09)	0.26 (0.06)	150	0.28 (0.03)	0.51 (0.09)	0.37 (0.03)
300	0.34 (0.04)	0.06 (0.02)	0.17 (0.04)	300	0.28 (0.03)	0.24 (0.05)	0.28 (0.03)
GAM				**RPART**			
75	0.16 (0.02)	0.31 (0.06)	0.24 (0.03)	75	0.22 (0.02)	0.58 (0.06)	0.44 (0.02)
150	0.15 (0.02)	0.21 (0.06)	0.18 (0.03)	150	0.17 (0.01)	0.55 (0.03)	0.31 (0.02)
300	0.17 (0.02)	0.08 (0.02)	0.13 (0.02)	300	0.09 (0.01)	0.63 (0.10)	0.22 (0.01)
MARS				**ACOSSO**			
75	0.15 (0.01)	0.89 (0.01)	0.55 (0.02)	75	0.27 (0.03)	0.46 (0.10)	0.34 (0.06)
150	0.09 (0.00)	0.93 (0.00)	0.31 (0.01)	150	0.26 (0.02)	0.32 (0.06)	0.27 (0.05)
300	0.10 (0.00)	0.92 (0.01)	0.19 (0.01)	300	0.06 (0.00)	0.76 (0.02)	0.13 (0.00)
Random Forest				**GBM**			
75	0.22 (0.03)	0.62 (0.14)	0.41 (0.06)	75	0.28 (0.03)	0.57 (0.14)	0.44 (0.08)
150	0.22 (0.03)	0.33 (0.06)	0.27 (0.05)	150	0.26 (0.04)	0.45 (0.08)	0.31 (0.06)
300	0.18 (0.01)	0.39 (0.04)	0.17 (0.02)	300	0.23 (0.04)	0.31 (0.08)	0.20 (0.02)
MLE GP				**MLE BGP**			
75	0.23 (0.02)	0.35 (0.04)	0.32 (0.04)	75	0.23 (0.02)	0.29 (0.07)	0.22 (0.01)
150	0.16 (0.01)	0.55 (0.02)	0.26 (0.02)	150	0.16 (0.02)	0.59 (0.08)	0.19 (0.01)
300	0.08 (0.00)	0.78 (0.03)	0.25 (0.01)	300	0.07 (0.02)	0.83 (0.07)	0.16 (0.00)

[a] Table structure same as in Table 4.

is an increasingly precise (small variance) estimate around the incorrect value. This leads to the poor coverages for larger sample sizes.

RPART has increasing coverage with sample size for y_3, but coverage only increases to 0.63 at $n = 300$. MARS once again has very good coverage at all sample sizes with decreasing CI length as n increases. ACOSSO has poor coverage on this example for small n, but increases to 0.76 for $n = 300$. The CI length and RMSE are higher than MARS for small sample sizes on this example, but length and RMSE are again the smallest of all the methods when $n = 300$.

Both Random Forest and GBM have decreasing coverage as sample size increases for y_3. The RMSE also does not go down as quickly for Random Forest and GBM as it does for the other methods.

The coverage for MLE GP is similar to ACOSSO in that coverage is poor for small n but increases to a reasonable level for $n = 300$. The decrease in RMSE is also similar to that for ACOSSO. A notable difference, however, is that the CI length for MLE GP does not go down nearly as much when moving from $n = 150$ to $n = 300$. MLE BGP is also similar to ACOSSO and MLE GP. MLE BGP has poor coverage for small n but is one of the best overall methods for y_3 with $n = 300$.

Lastly, Table 9 displays the results of the three sample sizes for y_3 for the unimportant inputs. This table shows results very similar to those for y_1 and y_2 on unimportant inputs for each of the methods.

Table 9: Results for y_3 averaged across the *unimportant* inputs with various sample sizes.

n	RMSE	Cover	Length	n	RMSE	Cover	Length
REG				**QREG**			
75	0.07 (0.00)	0.99 (0.00)	0.23 (0.00)	75	0.14 (0.00)	0.85 (0.01)	0.32 (0.00)
150	0.04 (0.00)	1.00 (0.00)	0.14 (0.00)	150	0.08 (0.00)	0.93 (0.00)	0.22 (0.00)
300	0.03 (0.00)	0.99 (0.00)	0.08 (0.00)	300	0.05 (0.00)	0.98 (0.00)	0.15 (0.00)
GAM				**RPART**			
75	0.02 (0.00)	0.90 (0.00)	0.05 (0.00)	75	0.08 (0.00)	0.96 (0.00)	0.23 (0.00)
150	0.01 (0.00)	0.91 (0.01)	0.04 (0.00)	150	0.04 (0.00)	0.98 (0.00)	0.14 (0.00)
300	0.01 (0.00)	0.92 (0.01)	0.03 (0.00)	300	0.02 (0.00)	0.95 (0.01)	0.06 (0.00)
MARS				**ACOSSO**			
75	0.06 (0.00)	0.98 (0.00)	0.23 (0.00)	75	0.06 (0.00)	0.93 (0.00)	0.19 (0.00)
150	0.04 (0.00)	0.99 (0.00)	0.11 (0.00)	150	0.03 (0.00)	0.98 (0.00)	0.10 (0.00)
300	0.02 (0.00)	0.97 (0.00)	0.08 (0.00)	300	0.01 (0.00)	0.99 (0.00)	0.02 (0.00)
Random Forest				**GBM**			
75	0.04 (0.00)	1.00 (0.00)	0.26 (0.00)	75	0.03 (0.00)	1.00 (0.00)	0.26 (0.00)
150	0.02 (0.00)	1.00 (0.00)	0.14 (0.00)	150	0.01 (0.00)	1.00 (0.00)	0.14 (0.00)
300	0.02 (0.00)	0.98 (0.00)	0.08 (0.00)	300	0.01 (0.00)	0.99 (0.00)	0.07 (0.00)
MLE GP				**MLE BGP**			
75	0.07 (0.00)	0.83 (0.00)	0.12 (0.00)	75	0.16 (0.00)	0.68 (0.00)	0.20 (0.00)
150	0.02 (0.00)	0.93 (0.00)	0.07 (0.00)	150	0.15 (0.00)	0.65 (0.00)	0.17 (0.00)
300	0.01 (0.00)	0.93 (0.01)	0.05 (0.00)	300	0.10 (0.00)	0.84 (0.00)	0.13 (0.00)

[a] Table structure same as in Table 5.

4.5 Summary of Simulation Results

It appears that the methods that require the strictest assumptions (QREG and GAM) can be among the best performers especially for small sample sizes. This is not surprising, as these assumptions are requiring less to be estimated from the data. Hence, a smaller sample is adequate for good estimation provided the assumptions imposed are not badly violated by the true function. As sample size is increased, some of the most flexible methods like GP, MARS, and ACOSSO can provide better estimators of the T_j, although this is not universally true. ACOSSO provided the best estimates of the T_j in terms of RMSE and narrow CIs when the sample size was large ($n = 300$) in all three examples. The CI coverage was reasonable in general for ACOSSO but was as low as 0.76 for y_3. MARS gave more consistent coverages for the CIs but also resulted in wider CIs and less accurate estimation (higher RMSE) than ACOSSO. Both methods seem to be useful and with complementary strengths. The MLE GP method also had good performance in general on all three examples. The RPART, Random Forest, GBM, and MLE BGP methods all performed well in certain cases, but were also very inconsistent in the results on the examples studied here. On this basis, we do not recommend using these methods since better options such as MARS, ACOSSO, and MLE GP exist.

The plot in Figure 8 gives the average RMSE for estimating T_j (averaged across important inputs) with each of the methods versus the required computing time to estimate T_j and produce CIs. Each method is plotted three times, one for each of the three output examples with $n = 300$. This plot indicates how much computing time is required relative to the accuracy of the individual methods. Specifically, ACOSSO, MLE GP, and MLE BGP take roughly an order of magnitude longer than the other methods. MARS was one of the better methods overall and is very fast to compute, making it an attractive option for use in practice. QREG also did very well in the first two examples. Given the ease of interpretability and speed of computation, QREG is another good option for practical use. Although ACOSSO and MLE GP take longer to compute than QREG and MARS, they are also consistently very good in terms of RMSE. Thus we recommend the use of these four methods (QREG, MARS, ACOSSO, MLE GP) to perform SA in actual analyses.

5 Practical Implementation

In this section we describe a simple yet effective strategy for the implementation of the approaches discussed thus far to carry out an actual sensitivity analysis on a computationally demanding model. We then provide an example sensitivity analysis using this strategy on a real data set from the 1996 WIPP compliance certification application (CCA).

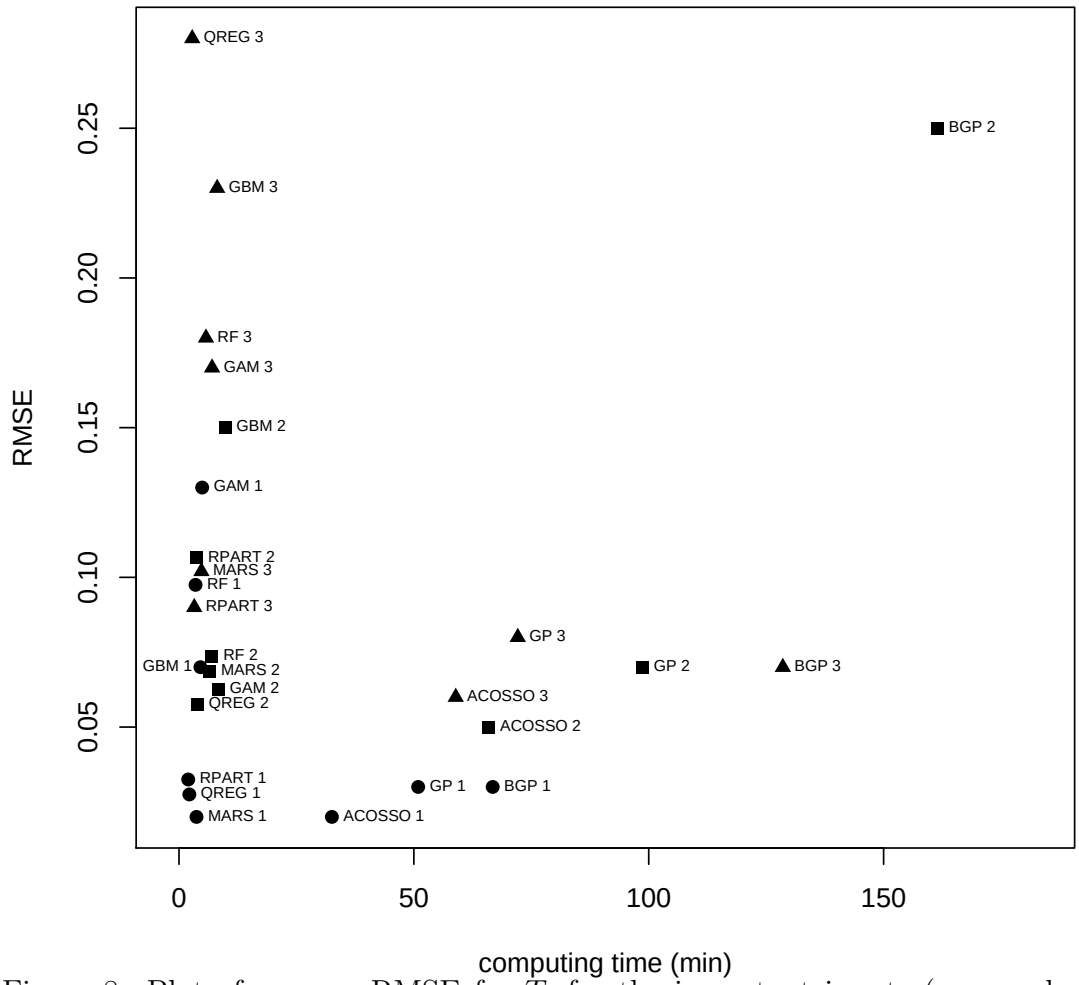


Figure 8: Plot of average RMSE for T_j for the important inputs (averaged within the three simulation examples) versus the average computing time (over the 100 realizations) needed for the method. Points are labeled by meta-model name and the output example number.

5.1 Algorithmic Description

Suppose that there are q outputs of interest in the analysis, $\boldsymbol{y} = y_1, \ldots, y_q$.

1. Determine appropriate distributions for each of the inputs to describe their uncertainty.

2. Take a sample (simple random, Latin hypercube, fixed design, etc.) from the input variables of size n.

3. Generate the outputs, $\boldsymbol{y}$, from the computer model at each of the n values of the input vector, $\boldsymbol{x}$.

Now repeat steps 4 and 5 for each of the q outputs.

4. Fit a rank regression to y_k. If the R^2 value of the fit is above R^2_{min}, then use the Standardized Rank Regression Coefficients (SRRCs) and Partial Rank Correlation Coefficients (PRCCs) to summarize input variable importance. Create bootstrap CIs for these quantities as described in Section 2.3 if desired.

5. If the R^2 value of the rank regression fit is below R^2_{min}, then fit several flexible regression methods. For each of these methods, calculate T_j based on the metamodel and the corresponding CIs if desired.

We recommend using four or five different flexible regression surfaces in step 5. It has been our experience that if rank regression does not sufficiently model the data, then it means the underlying surface is fairly complex. As such, more flexible methods will not always agree on their respective estimates of the T_j. Hence it is valuable to obtain several different estimates of varying complexity to gage how trustworthy these estimates are. We suggest using QREG, MARS, ACOSSO, and MLE GP for this purpose because (i) they cover a large spectrum of model complexity and continuity, and (ii) their performances seemed to complement each other fairly well in the simulations of Section 4.

The strategy above ensures that the more time consuming nonparametric methods are only used when they are needed. Some thought does need to go into the specification of the control parameter R^2_{min} however. This choice largely depends on the number of outputs, the time available for analysis, and the importance of the analysis, etc. We can only suggest that a reasonable guideline is to use R^2_{min} somewhere in the range of $[.75, .90]$. In the analysis presented in the next section, we use $R^2_{\text{min}} = .80$ for example.

5.2 Example Sensitivity Analysis

The data for the example presented here comes from an uncertainty/sensitivity analysis of a model for two phase fluid flow [57, 7, 58, 59] carried out as part of the 1996 CCA for the Waste Isolation Pilot Plant (WIPP) [7]. The CCA involved $p = 57$

uncertain variables, [60] with 31 of these variables used in the two-phase fluid flow analysis considered in this section; see Appendix A.1 for definitions of the variables. The two-phase fluid flow analysis considered six different scenarios (i.e., modeling cases) and generated several hundred time-dependent analysis results for each modeling case (i.e., see Table 1, Ref. [58], for a partial listing of these results). A small subset of these results is considered in this presentation. In particular, the modeling case corresponding to a drilling intrusion at 1,000 yr that penetrates both the repository and an underlying region of pressurized brine is used as an example (i.e., an E1 intrusion at 1,000 yr in the terminology of the 1996 WIPP CCA; see Table 6, [60]).

The example analysis used Latin hypercube sampling with sample size $n = 300$ to generate a mapping between analysis inputs and analysis results of the form Eq. (1.1). The time-dependent result *WAS_PRES* is analyzed at 1,000 yr and 10,000 yr. *WAS_PRES* is the Pressure (Pa) in the waste panel penetrated by a drilling intrusion (i.e., in the region corresponding to Cells 596-616 in Fig. 3 of [57]). The results at 1,000 yr are for undisturbed conditions immediately prior to the drilling intrusion at 1,000 yr. Because of this timing, the 1,000 yr results are unaffected by the drilling intrusion and thus are very different from the 10,000 yr results. The data for this illustration is available at *http//:www.stat.unm.edu/∼storlie/CompModSA/wipp_data.txt*. The input variables pairs (*HALPRM*, *HALCOMP*) and (*ANHPRM*, *ANHCOMP*) are very highly correlated, thus only *HALPRM* and *ANHPRM* were used in the presented SA.

To perform the SA on the two outputs considered here, we use the R statistical computing software. R is an open source software very similar to S-Plus. Go to *http://cran.r-project.org/* for documentation and more information on the the use of R. There is an R-package called 'CompModSA' available at `http://www.stat.unm.edu/ storlie` that has a function called `sensitivity` that can perform the procedures described in this paper.

Pressure at 1,000 yr (*WAS_PRES.1K)*

Table 10 gives the results of a SA performed on *WAS_PRES* at 1,000 years (*WAS_PRES.1K*) by following the steps described in Section 5.1. Notice that the R^2 value for rank regression is 0.94, so that rank regression is all that is performed on this output variable. These results indicate that the microbial degradation of cellulose (*WMICDFLG*) is clearly the most important variable affecting the Waste Pressure at 1,000 years with PRCC^2 between 0.89 and 0.95 as indicated by the CI. After *WMICDFLG*, corrosion rate for steel (*WGRCOR*) and the increase in brine saturation of waste due to capillary forces (*WASTWICK*) are also important variables affecting *WAS_PRES* at 1,000 years. *ANHPRM* may also have a small effect, with PRCC^2 values between 0.01 and 0.10. The remaining variables listed (*WGRMICI*, *ANHBCVGP*, *HALPRM*, *SHPRMSAP*) may have some small effect on *WAS_PRES* at 1,000 years, but do not have a statistically significant effect (at the 0.05 level of significance for example), as evidenced by p-values greater than 0.05 for each of these variables.

Table 10: Results for *WAS_PRES.1K* using Rank Regression and PRCCs

Estimated Model Summary: Rsq = 0.943, model df = 8

Input	Rsq [a]	SRRC [b]	$PRCC^2$ [c]	95% $PRCC^2$ CI [d]	p-val [e]
WMICDFLG	0.783	0.945	0.930	(0.893, 0.950)	0.000
WGRCOR	0.891	0.332	0.660	(0.578, 0.732)	0.000
WASTWICK	0.936	0.216	0.451	(0.334, 0.586)	0.000
ANHPRM	0.939	0.055	0.050	(0.007, 0.107)	0.020
WGRMICI	0.941	0.036	0.022	(0.000, 0.079)	0.200
ANHBCVGP	0.942	0.039	0.019	(0.000, 0.057)	0.080
HALPRM	0.943	0.028	0.013	(0.000, 0.080)	0.320
SHPRMSAP	0.943	0.027	0.012	(0.000, 0.050)	0.360

[a] Incremental (or cumulative) R^2 value of the rank regression model.

[b] Standardized rank regression coefficient (SRRC).

[c] Partial rank regression correlation coefficient squared ($PRCC^2$). This can be interpreted as the proportion of the remaining variance (left over in the rank data after including all of the other variables) that is explained by adding the current variable.

[d] CIs for $PRCC^2$ using the nonparametric bootstrap. Specifically, the procedure described in Section 2.3 is used with $\theta = PRCC^2$ and F^* equal to the empirical CDF.

[e] Bootstrap p-value for $H_0 : PRCC^2 = 0$ calculated according to Eq. (2.26) with $PRCC^2$ in place of T_j.

Pressure at 10,000 yr (WAS_PRES.10K)

Tables 11-14 give the SA results for WAS_PRES at 10,000 years. In this case, the $R^2 = 0.20$ for rank regression, so that the other more flexible meta models are then fit to the output. The estimates of the sensitivity index T_j of bore hole permeability (*BHPERM*) given by each of the four meta-models are between 0.44 and 0.70. Clearly *BHPERM* is the most influential input to this analysis but there is some discrepancy as to how much uncertainty is due to *BHPERM*.

The CIs from QREG (Table 11) indicate that *BHPERM* is responsible for 50% to 70% of the uncertainty in the Waste Pressure at 10,000 years. After this the variable importance rankings are less certain. However, the upper confidence limits suggest that brine pocket compressibility (*BPCOMP*) may account for a significant portion (upwards of 25%) of the uncertainty. Halite permeability (*HALPRM*), anhydrite permeability (*ANHPRM*), and *WGRCOR* all may account for a significant portion (16%, 12%, and 12% respectively) of the uncertainty as well. These results are consistent with those from ACOSSO (Table 13). ACOSSO indicates that halite porosity (*HALPOR*) may also account for as much as 12% of the uncertainty. MLE GP gives similar estimates and CIs as well (Table 14). However, a notable difference is that MLE GP estimates the percentage of the total uncertainty due to BHPERM to be somewhat higher (between 69% and 92%).

The CIs from MARS (Table 12) are a too wide to be of much use in this example. Several of the CI lengths are wider than 0.5. The CIs from the other methods are somewhat wide as well but they are more usable. In any case, the CIs described in this presentation are valuable especially when there is a moderate to large amount of variability in the estimates for T_j such as in this example. Namely, the wide CIs inform us not to blindly treat the problem as though $\hat{T}_j$ is the true value of T_j, but rather to rely on the CIs from QREG, ACOSSO, and MLE GP to guide any decision making.

6 Summary and Further Work

In this presentation, we have described several nonparametric regression methods that can be used as meta-models to calculate sensitivity measures. A bootstrap procedure for providing confidence intervals for sensitivity measures was also proposed and studied via simulation. These simulation examples indicated that simpler models such as quadratic regression and additive smoothing splines can work very well to estimate sensitivity measures in some cases, especially for small sample sizes. As sample size increases, more flexible models (MARS, ACOSSO, and MLE GP in particular) can provide better estimation. A practical guide for implementation of these techniques was also given.

For the purpose of SA, the meta-models that performed the best overall were MARS, ACOSSO, and MLE GP. ACOSSO and MLE GP tend to give narrower CIs for the total sensitivity indices, but MARS generally has better coverage for these

Table 11: Results for WAS_PRES.10K using Quadratic Regression.

Meta-model: QREG (see Sect. 2.3, Ref.[12])
Estimated Model Summary: Rsq = 0.880, model df = 89

Input	S_j [a]	S_{cum} [b]	$\hat{T}_j$ [c]	95% T_j CI [d]	p-val [e]
BHPERM	0.488	0.488	0.529	(0.495, 0.699)	0.000
BPCOMP	0.093	0.581	0.166	(0.134, 0.252)	0.000
HALPRM	0.119	0.700	0.104	(0.031, 0.164)	0.000
ANHPRM	0.083	0.784	0.094	(0.030, 0.122)	0.000
WGRCOR	0.039	0.823	0.078	(0.013, 0.117)	0.000
BPMAP	0.024	0.847	0.039	(0.000, 0.099)	0.140
HALPOR	0.051	0.897	0.038	(0.000, 0.087)	0.260
SHRGSSAT	0.013	0.911	0.036	(0.000, 0.079)	0.140
BPINTPRS	0.029	0.940	0.031	(0.000, 0.087)	0.200
ANHBCVGP	0.012	0.952	0.031	(0.000, 0.082)	0.180
SHPRNHAL	0.031	0.983	0.019	(0.000, 0.084)	0.460
WMICDFLG	0.017	1.000	0.017	(0.000, 0.093)	0.520

[a] Estimate given in Eq. (2.6) of the stepwise proportion of variance explained by the input and any of its interactions with inputs already in the model (i.e. those listed above the input in question).

[b] Cumulative proportion of variance explained by the input along with all inputs already in the model. This is the sum of the S_j up to and including the current input.

[c] The estimate given in Eq. (2.1) of the proportion of the total variance explained by the input and any of its interactions with other inputs.

[d] Bootstrap CI for T_j as given in Eq. (2.22).

[e] Bootstrap p-value for $H_0 : T_j = 0$ given in Eq. (2.26).

Table 12: Results for WAS_PRES.10K using MARS.[a]

Meta-model: MARS (see Sect. 3.1)
Estimated Model Summary: Rsq = 0.940, model df = 52

Input	S_j	S_{cum}	$\hat{T}_j$	95% T_j CI	p-val
BHPERM	0.407	0.407	0.546	(0.440, 0.937)	0.000
WGRCOR	0.086	0.493	0.158	(0.095, 0.313)	0.000
BPCOMP	0.083	0.576	0.154	(0.000, 0.747)	0.120
ANHPRM	0.092	0.668	0.127	(0.000, 0.691)	0.120
HALPRM	0.107	0.774	0.122	(0.000, 0.257)	0.040
BPMAP	0.048	0.823	0.050	(0.000, 0.130)	0.080
HALPOR	0.018	0.840	0.031	(0.000, 0.122)	0.280
WMICDFLG	0.018	0.858	0.025	(0.000, 0.056)	0.060
SHPRMDRZ	0.022	0.880	0.020	(0.000, 0.176)	0.180
WFBETCEL	0.003	0.883	0.020	(0.000, 0.164)	0.140
WGRMICH	0.005	0.888	0.019	(0.000, 0.316)	0.220
SALPRES	0.005	0.893	0.018	(0.000, 0.099)	0.300
WASTWICK	0.037	0.930	0.018	(0.000, 0.142)	0.220
WRBRNSAT	0.000	0.930	0.015	(0.000, 0.201)	0.380
BPINTPRS	0.025	0.955	0.012	(0.000, 0.079)	0.340

[a] Table structure same as in Table 11.

CIs. In difficult problems, however, such as the 29 input model with $n = 300$ of Section 5.2, MARS gives CIs too wide to be useful. ACOSSO and MLE GP also take much longer to compute (about 2 hours each on the example in Section 5.2 compared to about 10 minutes for MARS). Therefore a reasonable strategy is to first fit a rank regression model. If this does not provide an adequate fit, then fit a QREG model and a MARS model. If the MARS model gives CIs that are too wide to be useful, then fit an ACOSSO and/or a MLE GP model. This will minimize the use of the more expensive procedures.

In this presentation, the total sensitivity index, T_j, was used as an example to study the use of the proposed meta-model estimation and bootstrap CI approach. It is important to recognize that this same procedure will apply to any quantity of interest that requires a large number of computations of the output function f (e.g. other sensitivity measures and/or uncertainty measures like quantiles, threshold exceedence probabilities, etc.).

It is of interest to continually update this strategy as new meta-models are developed. For example, ACOSSO is very new but works quite well. There are also very recently developed GP models that provide variable selection that may be of use in this framework [52, 47, 53]. The bootstrap approach for CIs seems to work well on the test cases used here. However, it can be expensive to generate a large number of bootstrap samples depending on the method and sample size, etc. A possible alter-

Table 13: Results for WAS_PRES.10K using ACOSSO.[a]

Meta-model: ACOSSO (see Sect. 3.4)
Estimated Model Summary: Rsq = 0.920, model df = 110

Input	S_j	S_{cum}	$\hat{T}_j$	95% T_j CI	p-val
BHPERM	0.406	0.406	0.442	(0.391, 0.649)	0.000
HALPRM	0.108	0.514	0.124	(0.038, 0.155)	0.000
BPCOMP	0.115	0.629	0.124	(0.053, 0.205)	0.000
ANHPRM	0.106	0.735	0.097	(0.022, 0.152)	0.000
WGRCOR	0.015	0.750	0.052	(0.000, 0.110)	0.050
HALPOR	0.050	0.800	0.048	(0.000, 0.117)	0.200
BPMAP	0.039	0.840	0.045	(0.000, 0.095)	0.100
ANHBCVGP	0.013	0.852	0.041	(0.000, 0.109)	0.150
WRGSSAT	0.000	0.852	0.023	(0.000, 0.047)	0.050
ANRBRSAT	0.022	0.875	0.020	(0.000, 0.069)	0.150
SHPRNHAL	0.000	0.875	0.020	(0.000, 0.084)	0.275
SHRBRSAT	0.009	0.884	0.020	(0.000, 0.067)	0.225
SHBCEXP	0.000	0.884	0.014	(0.000, 0.063)	0.525
WMICDFLG	0.036	0.920	0.010	(0.000, 0.098)	0.775
SALPRES	0.011	0.930	0.010	(0.000, 0.044)	0.250

[a] Table structure same as in Table 11.

native would be to derive the asymptotic distribution of $\hat{T}_j$ as in [22] for some of the methods.

References

[1] W.L. Oberkampf, S.M. DeLand, B.M. Rutherford, K.V. Diegert, and K.F. Alvin. Error and uncertainty in modeling and simulation. *Reliability Engineering and System Safety*, 75:333–357, 2002.

[2] M.A. Christie, J. Glimm, J.W. Grove, D.M. Higdon, D.H. Sharp, and M.M. Wood-Schultz. Error analysis and simulations of complex phenomena. *Los Alamos Science*, 29:6–25, 2005.

[3] M.B. Beck. Water-quality modeling: A review of the analysis of uncertainty. *Water Resources Research*, 23:1393–1442, 1987.

[4] D.G. Cacuci. *Sensitivity and Uncertainty Analysis, Vol. 1: Theory.* Chapman and Hall/CRC Press, Boca Raton, FL, 2003.

[5] T. Turányi. Sensitivity analysis of complex kinetic systems. tools and applications. *Journal of Mathematical Chemistry*, 5:203–248, 1990.

Table 14: Results for WAS_PRES.10K using MLE GP.[a]

Meta-model: MLE GP (see Sect. 3.5)
Estimated Model Summary: Rsq = 0.995, model df = 246

Input	S_j	S_{cum}	$\hat{T}_j$	95% T_j CI	p-val
BHPERM	0.509	0.509	0.706	(0.692, 0.918)	0.000
BPCOMP	0.104	0.613	0.143	(0.069, 0.248)	0.000
HALPRM	0.125	0.737	0.123	(0.072, 0.182)	0.000
ANHPRM	0.119	0.856	0.093	(0.003, 0.156)	0.025
WGRCOR	0.052	0.909	0.068	(0.029, 0.124)	0.000
HALPOR	0.001	0.910	0.037	(0.000, 0.097)	0.125
SHPRMSAP	0.018	0.928	0.018	(0.000, 0.050)	0.175
BPMAP	0.015	0.943	0.016	(0.000, 0.082)	0.400
ANHBCVGP	0.007	0.950	0.015	(0.000, 0.054)	0.225
WRGSSAT	0.010	0.960	0.014	(0.000, 0.052)	0.250
SHPRMDRZ	0.000	0.960	0.013	(0.000, 0.045)	0.275
ANHBCEXP	0.000	0.960	0.012	(0.000, 0.046)	0.300
SHPRMCLY	0.004	0.964	0.012	(0.000, 0.052)	0.150
SHPRNHAL	0.003	0.967	0.011	(0.000, 0.044)	0.225
SHRBRSAT	0.000	0.967	0.010	(0.000, 0.049)	0.275

[a] Table structure same as in Table 11.

[6] J.C. Helton and R.J. Breeding. Calculation of reactor accident safety goals. *Reliability Engineering and System Safety*, 39(2):129–158, 1993.

[7] J.C. Helton and M.G. Marietta, editors. Special issue: The 1996 performance assessment for the Waste Isolation Pilot Plant. *Reliability Engineering and System Safety*, 69(1-3):1–451, 2000.

[8] A. Saltelli, Chan K., and E.M. Scott, editors. *Sensitivity Analysis*. New York, NY: Wiley, 2000.

[9] J.C. Helton, J.D. Johnson, J.D. Sallaberry, and C.B. Storlie. Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering and System Safety*, 91:1175–1209, 2006.

[10] R.L. Iman and J.C. Helton. An investigation of uncertainty and sensitivity analysis techniques for computer models. *Risk Analysis*, 8:71–90, 1988.

[11] J.E. Oakley and A. O'Hagan. Probabilistic sensitivity analysis of complex models: A Bayesian approach. *Journal of the Royal Statistical Society: Series B*, 66:751–769, 2004.

[12] C.B. Storlie and J.C. Helton. Multiple predictor smoothing methods for sensitivity analysis: Description of techniques. *Reliability Engineering and System Safety*, 93:28–54, 2007.

[13] J.C. Helton and F.J. Davis. Latin hypercube sampling and the propagation of uncertainty in analysis of complex systems. *Reliability Engineering and System Safety*, 81:23–69, 2003.

[14] M.D. McKay, Beckman R.J., and W.J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21:239–245, 1979.

[15] R.L. Iman and W.J. Conover. The use of the rank transform in regression. *Technometrics*, 21:499–509, 1979.

[16] J.C. Helton and F.J. Davis. Sampling based methods. In A. Saltelli, K. Chan, and E.M. Scott, editors, *Sensitivity Analysis*, pages 101–153. New York, NY: Wiley, 2000.

[17] J.P.C. Kleijnen and J.C. Helton. Statistical analysis of scatterplots to identify important factors in large-scale simulations, 1: Review and comparison of techniques. *Reliability Engineering and Systems Safety*, 65:147–185, 1999.

[18] T. Homma and A. Saltelli. Importance measures in global sensitivity analysis of model output. *Reliability Engineering and System Safety*, 52:1–17, 1996.

[19] SAS institute. *SAS/STAT User's Guide*. http://v8doc.sas.com/sashtml/stat/index.htm, 2008.

[20] A.C. Davison and D.V. Hinkley. *Bootstrap Methods and their Application*. New York, NY: Cambridge University Press, 1997.

[21] W. Härdle. *Applied Nonparametric Regression*. New York, NY: Cambridge University Press, 1990.

[22] K. Doksum and A. Samarov. Nonparametric estimation of global functionals and a measure of the explanatory power of covariates in regression. *Annals of Statistics*, 23:1443–1473, 1995.

[23] J.H. Friedman. Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19:1–141, 1991.

[24] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[25] J.H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.

[26] C.B. Storlie, H.D. Bondell, B.J. Reich, and H.H. Zhang. Surface estimation, variable selection, and the nonparametric oracle property. *Journal of the Royal Statistical Society: Series B*, 2008. Submitted for publication. URL: www.stat.unm.edu/∼storlie/acosso.pdf.

[27] J. Sacks, W.J. Welch, T.J. Mitchel, and H.P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–435, 1989.

[28] M.C. Kennedy and A. O'Hagan. Bayesian calibration of computer models (with discussion). *Journal of the Royal Statistical Society: Series B*, 63:425–464, 2001.

[29] T. Hastie, R.J. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, NY, 2001.

[30] R. Schapire. Strength of weak learnability. *Journal of Machine Learning*, 5:197–227, 1990.

[31] Y. Freund. Boosting a weak learning algorithm by majority. *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 202–216, 1990.

[32] P. Craven and G. Wahba. Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerical Mathematics*, 31:377–403, 1979.

[33] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 2:1137–1143, 1995.

[34] R.L. Eubank. *Nonparametric Regression and Spline Smoothing*. CRC Press, 1999.

[35] T. Hastie and R.J. Tibshirani. *Generalized Additive Models*. Chapman & Hall/CRC, 1990.

[36] A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Norwell, MA: Kluwer Academic Publishers, 2004.

[37] G. Wahba. *Spline Models for Observational Data*. CBMS-NSF Regional Conference Series in Applied Mathematics, 1990.

[38] M. Schimek, editor. *Smoothing and Regression: Approaches, Computation, and Application*. John Wiley & Sons, Inc., New York, NY, 2000.

[39] C. Gu. *Smoothing Spline ANOVA Models*. Springer-Verlag, New York, NY, 2002.

[40] Y. Lin and H. Zhang. Component selection and smoothing in smoothing spline analysis of variance models. *Annals of Statistics*, 34:2272–2297, 2006.

[41] R.J. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58:267–288, 1996.

[42] G. Matheron. Principles of geostatistics. *Economic Geology*, 58:1246–1266, 1963.

[43] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Upper Saddle River, NJ: Pearson Prentice Hall, 6th edition, 2007.

[44] M.L. Stein. *Interpolation of Spatial Data*. Springer-Verlag, New York, NY, 1999.

[45] R.B. Gramacy and H.K. Lee. Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 2008. to appear.

[46] C.E Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2005.

[47] C. Linkletter, D. Bingham, N. Hengartner, D. Higdon, and K. Ye. Variable selection for Gaussian process models in computer experiments. *Technometrics*, 48:478–490, 2006.

[48] W.J. Welch, R.J. Buck, J. Sacks, H.P. Wynn, T.J. Mitchell, and M.D. Morris. Screening, predicting, and computer experiments. *Technometrics*, 34:15–25, 1992.

[49] G. Dancik. mlegp: an R package for Gaussian process modeling and sensitivity analysis. *http://cran.r-project.org/web/packages/mlegp/vignettes/gp.pdf*, 2007.

[50] N.A.C. Cressie. *Statistics for Spatial Data.* New York, NY: John Wiley & Sons, 1993.

[51] A. O'Hagan. Bayesian analysis of computer code outputs: a tutorial. *Reliability Engineering and System Safety*, 91:1290–1300, 2006.

[52] B.J. Reich, C.B. Storlie, and H.D. Bondell. Bayesian variable selection for non-parametric regression models. *Technometrics*, 2007. Submitted for publication.

[53] A. Marrel, B. Iooss, F. Van Dorpe, and E. Volkova. An efficient methodology for modelling complex computer codes with Gaussian processes. *Computational Statistics and Data Analysis*, 52:4731–4744, 2008.

[54] F. Campolongo, A. Saltelli, T. Sorensen, and S. Tarantola. Hitchhiker's guide to sensitivity analysis. *Sensitivity Analysis*, 2000. Ed. A. Saltelli, K. Chan, and M. Scott. New York, NY: John Wiley & Sons.

[55] J.C. Helton and F.J. Davis. Illustration of sampling-based methods for uncertainty and sensitivity analysis. *Risk Analysis*, 22:591–622, 2002.

[56] C.B. Storlie and J.C. Helton. Multiple predictor smoothing methods for sensitivity analysis: Example results. *Reliability Engineering and System Safety*, 93:55–77, 2007.

[57] P. Vaughn, J.E. Bean, J.C. Helton, M.E. Lord, R.J. MacKinnon, and J.D. Schreiber. Representation of two-phase flow in the vicinity of the repository in the 1996 performance assessment for the Waste Isolation Pilot Plant. *Reliability Engineering and System Safety*, 69:205–226, 2000.

[58] J.C. Helton, J.E. Bean, K. Economy, J.W. Garner, R.J. MacKinnon, J. Miller, J.D. Schreiber, and P. Vaughn. Uncertainty and sensitivity analysis for two-phase flow in the vicinity of the repository in the 1996 performance assessment for the Waste Isolation Pilot Plant: Undisturbed conditions. *Reliability Engineering and System Safety*, 69(1-3):227–261, 2000.

[59] J.C. Helton, J.E. Bean, K. Economy, J.W. Garner, R.J. MacKinnon, J. Miller, J.D. Schreiber, and P. Vaughn. Uncertainty and sensitivity analysis for two-phase flow in the vicinity of the repository in the 1996 performance assessment for the Waste Isolation Pilot Plant: Disturbed conditions. *Reliability Engineering and System Safety*, 69(1-3):263–304, 2000.

[60] J.C. Helton, M.A. Martell, and M.S. Tierney. Characterization of subjective uncertainty in the 1996 performance assessment for the Waste Isolation Pilot Plant. *Reliability Engineering and System Safety*, 69(1-3):191–204, 2000.

A Definition of Variables involved in Two-Phase Fluid Flow Example

A.1 Input Variables

Listed below are the input variables considered in example sensitivity analyses for Two-Phase Fluid Flow in Section 5.2 (Source: Table 1, Ref. [60])

ANHBCEXP - Brooks-Corey pore distribution parameter for anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 0.491 to 0.842. Mean, Median: 0.644, 0.644.

ANHBCVGP - Pointer variable for selection of relative permeability model for use in anhydrite. Distribution: Discrete with 60% 0, 40% 1. Value of 0 implies Brooks-Corey model; value of 1 implies van Genuchten-Parker model.

ANHCOMP - Bulk compressibility of anhydrite (Pa-1). Distribution: Student's with 3 degrees of freedom. Range: 1.09×10^{-11} to 2.75×10^{-10} Pa^{-1}. Mean, Median: 8.26×10^{-11} Pa^{-1}, 8.26×10^{-11} Pa^{-1}. Correlation: -0.99 rank correlation with ANHPRM.

ANHPRM - Logarithm of anhydrite permeability (m^2). Distribution: Student's with 5 degrees of freedom. Range: -21.0 t o -17.1 (i.e., permeability range is $1 \times 10 - 21$ to $1 \times 10 - 17.1$ m^2). Mean, Median: -18.9, -18.9. Correlation : -0.99 rank correlation with ANHCOMP.

ANRBRSAT - Residual brine saturation in anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 7.85×10^{-3} to 1.74×10^{-1}. Mean, Median: 8.36×10^{-2}, 8.36×10^{-2}.

ANRGSSAT - Residual gas saturation in anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 1.39×10^{-2} to 1.79×10^{-1}. Mean, median: 7.71×10^{-2}, 7.71×10^{-2}

BHPRM - Logarithm of borehole permeability (m2). Distribution: Uniform. Range: -14 to -11 (i.e., permeability range is 1×10^{-14} to 1×10^{-11} m^2). Mean, median: -12.5, -12.5.

BPCOMP - Logarithm of bulk compressibility of brine pocket (Pa^{-1}). Distribution: Triangular. Range: -11.3 to -8.00 (i.e., bulk compressibility range is $1 \times 10^{-11.3}$ to 1×10^{-8} Pa^{-1}). Mean, mode: -9.80, -10.0. Correlation: -0.75 rank correlation with BPPRM.

BPINTPRS - Initial pressure in brine pocket (Pa). Distribution: Triangular. Range: 1.11×10^7 to 1.70×10^7 Pa. Mean, mode: 1.36×10^7 Pa, 1.27×10^7 Pa.

BPPRM - Logarithm of intrinsic brine pocket permeability (m^2). Distribution: Triangular. Range: -14.7 to -9.80 (i.e., permeability range is $1 \times 10{-}^{14.7}$ to $1 \times 10^{-}9.80$ m^2). Mean, mode: -12.1, -11.8. Correlation: -0.75 rank correlation with BPCOMP.

BPVOL - Pointer variable for selection of brine pocket volume. Distribution: Discrete, with integer values 1, 2, ..., 32 equally likely.

HALCOMP - Bulk compressibility of halite (Pa^{-1}). Distribution: Uniform. Range: 2.94×10^{-12} to 1.92×10^{-10} Pa^{-1}. Mean, median: 9.75×10^{-11} Pa^{-1}, 9.75×10^{-11} Pa^{-1}. Correlation: -0.99 rank correlation with HALPRM.

HALPOR - Halite porosity (dimensionless). Distribution: Piecewise uniform. Range: 1.0×10^{-3} to 3×10^{-2}. Mean, median: 1.28×10^{-2}, 1.00×10^{-2}.

HALPRM - Logarithm of halite permeability (m^2). Distribution: Uniform. Range: -24 to -21 (i.e., permeability range is 1×10^{-24} to 1×10^{-21} m^2). Mean, median: -22.5, -22.5. Correlation: -0.99 rank correlation with HALCOMP.

SALPRES - Initial brine pressure, without the repository being present, at a reference point located in the center of the combined shafts at the elevation of the midpoint of Marker Bed (MB) 139 (Pa). Distribution: Uniform. Range: 1.104×10^7 to 1.389×10^7 Pa. Mean, median: 1.247×10^7 Pa, 1.247×10^7 Pa.

SHBCEXP - Brooks-Corey pore distribution parameter for shaft (dimensionless). Distribution: Piecewise uniform. Range: 0.11 to 8.10. Mean, median: 2.52, 0.94.

SHPRMSAP - Logarithm of permeability (m2) of asphalt component of shaft seal (m2). Distribution: Triangular. Range: -21 to -18 (i.e., permeability range is 1×10^{-21} to 1×10^{-18} m2). Mean, mode: -19.7, -20.0.

SHPRMCLY - Logarithm of permeability (m2) for clay components of shaft. Distribution: Triangular. Range: -21 to -17.3 (i.e., permeability range is 1×10^{-21} to $1 \times 10^{-17.3}$ m2). Mean, mode: -18.9, -18.3.

SHPRMCON - Same as SHPRMASP but for concrete component of shaft seal for 0 to 400 yr. Distribution: Triangular. Range: -17.0 to -14.0 (i.e., permeability range is 1×10^{-17} to 1×10^{-14} m2). Mean, mode: -15.3, 15.0.

SHPRMDRZ - Logarithm of permeability (m2) of DRZ surrounding shaft. Distribution: Triangular. Range: -17.0 to -14.0 (i.e., permeability range is 1×10^{-17} to 1×10^{-14} m2). Mean, mode: -15.3, -15.0.

SHPRMHAL - Pointer variable (dimensionless) used to select permeability in crushed salt component of shaft seal at different times. Distribution: Uniform. Range: 0 to 1. Mean, mode: 0.5, 0.5. A distribution of permeability (m2) in the crushed salt component of the shaft seal is defined for each of the following time intervals: [0, 10 yr], [10, 25 yr], [25, 50 yr], [50, 100 yr], [100, 200 yr], [200, 10000 yr]. SHPRMHAL is used to select a permeability value from the cumulative distribution function for permeability for each of the preceding time intervals with result that a rank correlation of 1 exists between the permeabilities used for the individual time intervals.

SHRBRSAT - Residual brine saturation in shaft (dimensionless). Distribution: Uniform. Range: 0 to 0.4. Mean, median: 0.2, 0.2.

SHRGSSAT - Residual gas saturation in shaft (dimensionless). Distribution: Uniform. Range: 0 to 0.4. Mean, median: 0.2, 0.2.

WASTWICK - Increase in brine saturation of waste due to capillary forces (dimensionless). Distribution: Uniform. Range: 0 to 1. Mean, median: 0.5, 0.5.

WFBETCEL - Scale factor used in definition of stoichiometric coefficient for microbial gas generation (dimensionless). Distribution: Uniform. Range: 0 to 1. Mean, median: 0.5, 0.5.

WGRCOR - Corrosion rate for steel under inundated conditions in the absence of CO2 (m/s). Distribution: Uniform. Range: 0 to 1.58×10^{-14} m/s. Mean, median: 7.94×10^{-15} m/s, 7.94 × 10−15 m/s.

WGRMICH - Microbial degradation rate for cellulose under humid conditions (mol/kg·s). Distribution: Uniform. Range: 0 to 1.27×10^{-9} mol/kg·s. Mean, median: 6.34×10^{-10} mol/kg·s, 6.34×10^{-10} mol/kg·s.

WGRMICI - Microbial degradation rate for cellulose under inundated conditions (mol/kg·s). Distribution: Uniform. Range: 3.17×10^{-10} to 9.51×10^{-9} mol/kg·s. Mean, median: 4.92×10^{-9} mol/kg·s, 4.92×10^{-9} mol/kg·s.

WMICDFLG - Pointer variable for microbial degradation of cellulose. Distribution: Discrete, with 50% 0, 25% 1, 25% 2. WMICDFLG = 0, 1, 2 implies no microbial degradation of cellulose, microbial degradation of only cellulose, microbial degradation of cellulose, plastic, and rubber.

WRBRNSAT - Residual brine saturation in waste (dimensionless). Distribution: Uniform. Range: 0 to 0.552. Mean, median: 0.276, 0.276.

WRGSSAT - Residual gas saturation in waste (dimensionless). Distribution: Uniform. Range: 0 to 0.15. Mean, median: 0.075, 0.075.